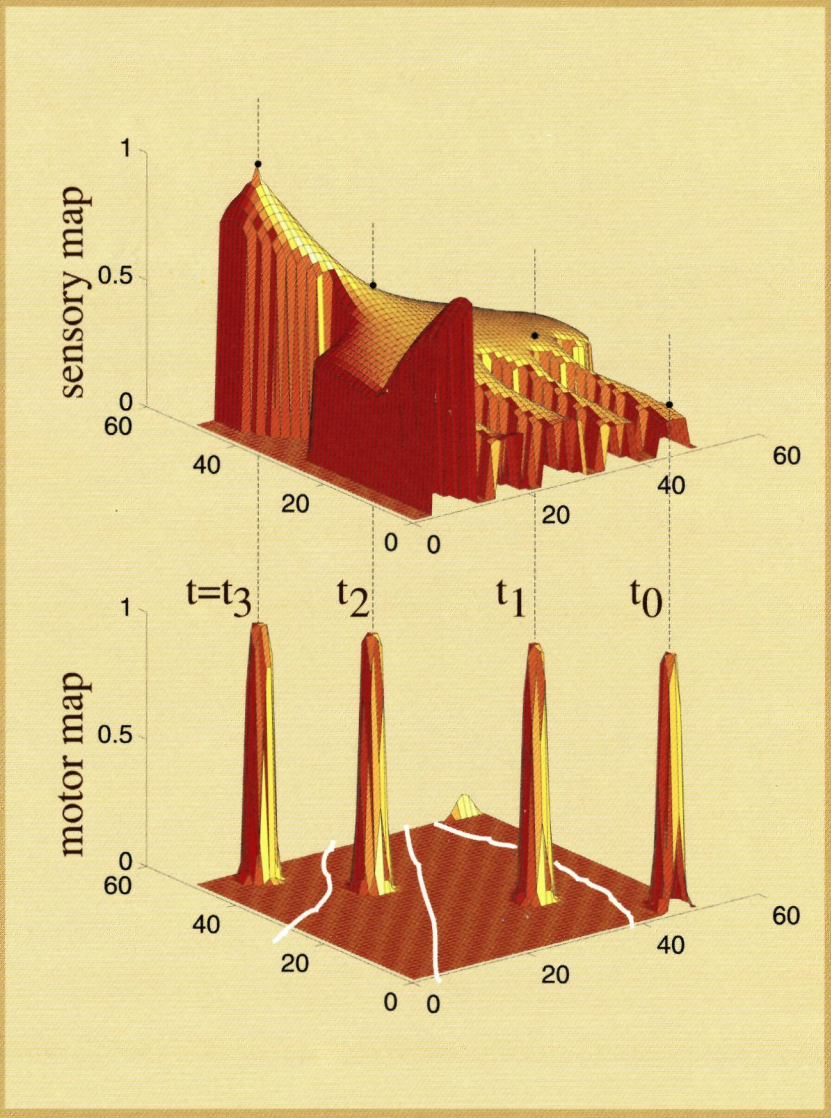
TRAJECTORY FORMATION AND POPULATION CODING WITH TOPOGRAPHICAL NEURAL NETWORKS



Roy Glasius

Trajectory Formation and Population Coding with Topographical Neural Networks

een wetenschappelijke proeve op het gebied van
de Natuurwetenschappen

Proefschrift

ter verkrijging van de graad van doctor
aan de Katholieke Universiteit Nijmegen,
volgens besluit van het College van Decanen
in het openbaar te verdedigen
op Maandag 24 november 1997
des namiddags om 1.30 uur precies

door

Roy Glasius

geboren op 15 maart 1961 te Bandung

Promotor: Prof. Dr. C.C.A.M. Gielen
Manuscriptcommissie: H.J. Kappen, B.J.A. Krose

This work was supported by the Dutch Foundation for Biophysics (NWO) and by the Dutch Foundation for Neural Networks (SNN).

CIP-GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Glasius, Roy

Trajectory Formation and Population Coding with Topographical Neural Networks
Roy Glasius - [S.l. : s.n.]. - Ill.
Proefschrift Nijmegen. - Met lit. opg. - Met samenvatting.
ISBN 90-9010985-4
Trefw.: Neural Network-Trajectory formation-Robotics.

About the cover:
The figure on the cover illustrates a result of the trajectory formation model (described in chapter four), where it was used to control movements of a two link robot arm in the middle of five obstacles. The top half of the picture shows the activation landscape of the sensory map. The target position corresponds to the sensory neuron with the largest activation, i.e. at the top of the hill. The bottom half of the picture shows parts of the motor map, each one at a different time. A cluster of activation smoothly and continuously shifts to the target position, never overlapping obstacle positions. The cluster follows the path described by the steepest ascent on the hill of the sensory map.

Contents

Chapter 1

Introduction and Summary

Despite of the enormous progress in artificial intelligence, artificial intelligence can in no way compete yet with natural intelligence, which can be shortly defined as the behavior of biological organisms in the various conditions of a normal biotope. It is especially in noisy, fuzzy, ill-defined conditions (which are typical for a natural environment) that the performance of artificial systems is dramatically poor. In this context humans and animals are remarkable creatures. With small effort they are capable to analyze a visual scene, to recognize objects of interest, to select a goal for a movement and to orient and to move around in a cluttered and dynamic environment reaching their goal without collisions. For example, it is amazing to see how many people can be allowed to enter a department store before people start bumping on each other.

To be aware of the changes in the environment man possesses a large number of sensors, all generating a continuous flow of afferent neural signals. These signals provide a huge stream of information to the central nervous system. Take for example the task "walking". Involved are an enormous number of sensory neurons inside the eyes, the vestibular organ, tactile receptors in the skin and in muscles and each one projects to specific regions in the central nervous system. These regions each have a large number of neurons and are complexly interconnected to each other and to other areas that are involved in the coordination of muscles, e.g. in the process of walking. Timing in this matter is essential and therefore some information streams are of a cyclical nature. In this view such an easy task as walking in a department store becomes a masterpiece of complex process control. The central nervous system is without question a very powerful and complex system which has inspired generations of researchers in trying to find a better understanding and which will inspire many more.

In addition to a better understanding of the fundamental processes underlying brain function in biological systems, the results of scientific research on brain function may also result in the implementation of specific brain functions in artificial applications. An example is the development of neural prostheses, which take over from deficient neurophysiological systems (for example, there are many persons on earth who have had implanted an artificial cochlea, and the development of an artificial retina is making good progress). In addition, these results can be used in robots and artificial intelligence to assist human performance.

In this thesis we will develop some neural network models which simulate trajectory planning and obstacle avoidance. The core of the problem is that a trajectory has to be found which brings the subject or robot from an initial position to a target position. The subject or robot has to find a feasible path in a complex environment in such a way that it reaches the target position without collisions with static and moving obstacles. In general, the path does not have to be the shortest path. For example, in order to avoid collisions, the path will be chosen such that the subject or robot stays at some distance from the objects in the environment. Moreover, the criterion of "the shortest path" would require sharply curved trajectories, which is a rather inefficient way of moving. Also, some cost functions may be involved. For example, it may be "more expensive" to take the shortest path (in Euclidean space) if it it across a steep mountain, than to take a longer path throughout the valley.

In the literature this problem is referred to as "Path Planning" or "Trajectory formation". Path Planning assumes prior knowledge about the "where" and "when" of objects in the environment. The path planner then calculates a series of feasible paths from the initial position to reach the goal, taking into account the positions of obstacles on a certain point in time. Any changes in the environment are not incorporated during the planning process. With "Trajectory Formation" the system incorporates the continuous changes in the environment during path planning. This is a continuous process at which at any moment in time, depending on the positions of the moving objects, a new movement path has to be generated. In the end, if possible, the (moving) target has to be reached. "Path Planning" and "Trajectory Formation" play an important role in many applications, such as: controlling autonomous robots and robot arms, finding an optimal routing on a chip or a printed circuit board, and with traffic navigation.

In the early days, sequential algorithms were developed to calculate a path for an autonomous moving robot in a static environment [4, 12, 35, 41, 47, 76]. Most of the algorithms subdivided the free workspace into subspaces and constructed a

graph of all possible movements, which connect the initial position to the target position. In the next step, for every solution, i.e. for every path between the initial position and the target position, the length of the path was calculated. In the end the shortest path was chosen to be the optimal solution. This brute force method contained some serious drawbacks. When a higher resolution of free subspaces was chosen, a combinatorial explosion of possible solutions takes place, resulting in unrealistic computing time. An additional disadvantage was, that the environment was not allowed to change, since the time consuming construction of the graph and the extensive search did not allow flexible real-time generation of the complex graphs.

Next, the "potential field" method was developed [36, 42], which was able to perform in a dynamic environment. The basic idea behind this method is the following. To each object an artificial charge is assigned. The target receives an attracting charge and the obstacles have repulsive charges. By using the equations from electrodynamics, the algorithm is able to calculate the forces and the resulting movements of the robot in the potential field. This approach generates a path which is continuous and smooth. However, despite some attempts to change the model [3, 5, 11, 53, 75], it frequently occurred that the algorithm is trapped in local minima of the potential field (see figure 1.1). The local minima are the result of the sometimes complex configuration of repulsive objects and the attracting target. In such a case the only possible solution requires that the robot moves in a direction opposite to the potential gradient before it can move to the target.

Contrary to the "potential field" method, the "wave propagation" or the "distance transform" method [13, 47, 33, 58, 66] is always able to find the proper solution, if such a solution exists. In this model the complete workspace is subdivided in small subspaces. Each subspace is represented by a node in a graph, and connected nodes represent neighboring subspaces. The node corresponding to the subspace that contains the target position, passes a signal to those neighboring nodes that have no obstacle position in its subspace. In the next step, these neighbors send a signal to their obstacle-free neighbors, etc. Every time a node passes a signal to other nodes, its activation value is incremented. The signals spread over the nodes in a wave-like manner always avoiding the nodes which correspond to the obstacles. As soon as the node that corresponds to the current position receives a signal, the process terminates. Subsequently, starting from the last node the algorithm generates a chain of movements, each time to the neighboring node with the largest activation. In the end, if a solution exists, the target is reached. This model generates a trajectory by the chain of via-points.

The above mentioned path planners were implemented as computer algo-

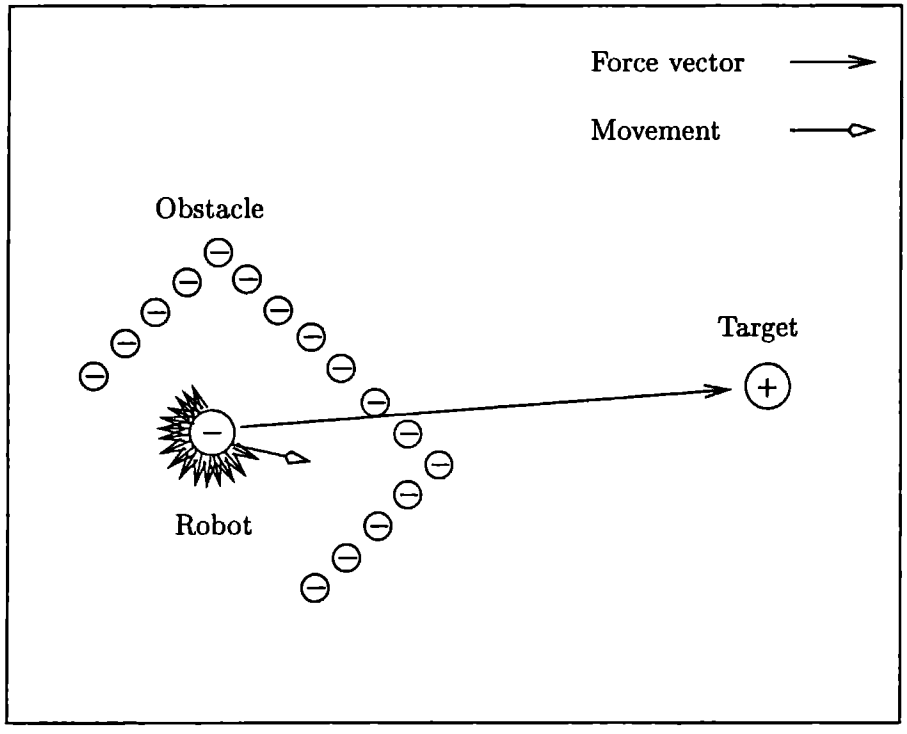


Figure 1.1: In the "potential field" method only the robot is allowed to move according the laws of electrodynamics. The obstacle consists of repulsive negative charged elements. The target has an attracting positive charge. The movement is the result of all the attracting and repelling forces (arrows) in the system. In this case the robot will be trapped inside the concave obstacle.

rithms which were executed on a sequential processor. It is clear that these methods contribute little to a better understanding of the possible processes in the brain, which acts as a parallel processor with many neurons. Many questions are left open. Is it possible to interpret neural activities? If possible, in what kind of coordinate system are the environment, the obstacles and the target represented in the brain? How does the brain calculate a path? How does it control the many muscles which are involved in movements ? Is it possible to translate the principles of the above mentioned sequential algorithms in terms of neural structures?

Although a number of basic principles of the working of the brain have already been unraveled, a working model of a plausible neural path planner does not exist. In this thesis we want to contribute to fill up this gap to some extent and we will provide a number of explicit neural network models for path planning. The models are based upon biologically plausible principles on brain function together with some additional assumptions.

Our models are described in terms of neural networks which consist of: artificial neurons, the connections between the neurons and the dynamics of the neural activities. Each neuron is a simple processing unit. It integrates all incoming signals, processes these according to a simple formula and then sends this output signal to other neurons. It is assumed that the neural network receives information about the objects in its environment from some special sensor-neurons. To represent its environment internally the neural network has subdivided the entire workspace in many small subspaces, like the receptive fields of sensory neurons.

When the neurons in the network are arranged such that adjacent neurons in the network have adjacent receptive fields in workspace, the neural network is called a topologically ordered map (see figure 1.2). This topographic ordering can be the result of some autonomous adaptive learning process of the connections based on the correlation between sensory information and neural activity within the neural map. This adaptive process can be described by the so-called "learning rules". These learning rules are described by models such as that by Kohonen [39, 59] or by the "vector quantization" methods [59, 17, 49].

From the literature it is well known, that neurons do not react in complete isolation. Usually a large number of neurons responds to a single stimulus. The responding neurons have, more or less, similar receptive field properties and the receptive fields are located in each others neighborhood. The problem to which many researchers in neurophysiology are confronted with is, how to interpret the activity of a population of active neurons in terms of sensory of motor events. Georgopoulos et al. [19] proposed a model which, based upon experimental results, gives an interpretation of the neural activities of such a population of neurons in the motor cortex in terms of movements of a limb. The estimate of movement direction is obtained by summation of the preferred directions of all neurons in motor cortex (i.e. the movement direction of the cell for which the cell gives the largest activity) weighted by the activity of the cell. The result is called the "Population Vector".

The population vector provides a good estimate of the sensory or motor events, coded by the neuronal activity as long as the sensory or motor space is sampled by the neuronal population in a uniform way. However, it is well known, that the distribution of neurons does not give a uniform sampling of the

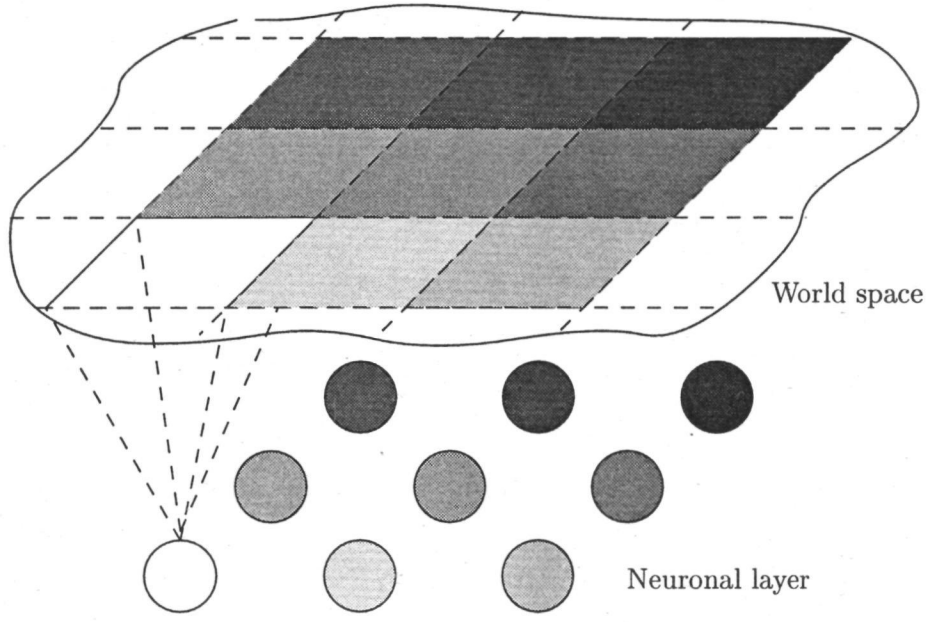


Figure 1.2: The world space is divided in subspaces. These are the receptive fields of the neurons. The neural network consists of a layer of neurons. The receptive fields are associated to their corresponding neurons by color (grey scale). In this case the neural layer is called a "Topographical map" because the order of receptive fields in the world space is the same as the order of the corresponding neurons in the neuronal layer.

sensory or motor space. Straight forward use of the population vector in these cases would result in a biased estimate. However, it is possible to extend the idea of Georgopoulos, in particular the theory of the population vector of Seung and Sompolinsky [68], to the use for non-uniformly distributed maps provided that certain requirements with respect to the population coding are met. In chapter five we will study the "Population Vector" in the case of non-uniformly distributed neural networks. A result of this study is that our trajectory formation model can be used with non-uniformly distributed topological neural networks.

A coding similar to the population vector is used for two neural network models described in chapter two. We assume that the population of active neurons changes continuously as a result of the intrinsic dynamics of the network. The two neural network models are the analogon of the potential algorithm. The first one is described with binary (two-valued) neurons with a stochastic behavior, the second one with continuous neurons and a deterministic behavior.

By means of excitatory short-range and inhibitory long-range connections between the neurons, a stable coherent cluster of neural activations is obtained in the map. Connections of external sensors to the neurons in the map are established with a strength inversely proportional to the distance between the receptive field of the sensory neuron and the receptive field of the neuron in the map. All neurons are excited except for those that code the position of obstacles, which are fully inhibited. Due to the sensory input about target position the cluster of activity in the map moves gradually in the direction of the target. This model, like the potential field method, suffers from local minima in which it can be trapped.

The neural network analog of the wave propagation method is the subject of chapter three. As before we use a topologically ordered neural network. This time we only use the short range excitatory connections between the neurons. External connections provide excitatory input only to the neuron which has a receptive field corresponding to that of the target neuron and fully inhibit neurons that code the position of obstacles. When the target neuron is activated by means of the external input, it will activate its neighbors, i.e. those neurons which are connected to it. If the strength of the short range connections are chosen properly, a wave of activation travels over the neural map. In this process the activation values decrease with the number of "passing" neurons in the chain. In the end an activation landscape emerge, shaped as a hill and therefore from each point on the map an optimal path to the target (top) can be constructed through successive jumps to those neurons that have the largest activation. Note that this last procedure requires a supervisor or homunculus, which is not realistic for biological neural networks. However, the method performs quite well in the sense that it always generates a good trajectory.

Eventually, we succeed in our objective using the model presented in chapter four in a biologically plausible way. In this model we combined the two previously mentioned models into a two-layered feed forward neural network. The network analogue to the wave propagation model is the first (input) layer which creates, as a result of external excitatory input, an activation landscape where the top of the activation corresponds to the target position. The neurons in this first map are connected one-to-one to neurons in a second layer. The second network (motor

layer) behaves like a potential field neural network which has a convex cluster of active neurons in its topological map. By means of an inhibitory connection between each neuron in the first layer to the corresponding neuron in the second layer, the cluster will shift continuously and smoothly along the optimal path to the target position.

Each chapter contains computer simulations as an illustration of the model that is studied in that chapter.

An additional quality of the last model is that the generated path is continuous and smooth. As a consequence, the motor system is able to operate the robot continuously and smoothly. Should the trajectory formator only generate so called "via-points", then an additional module is needed to mediate between the trajectory generator and the motor system. This interface has to prevent the robot to generate motions with jerks and sharp curves when moving along the via-points.

Because of the small number of connections between neurons in the last model and because of the fact that both the dynamics of the neurons and the nonlinear input-output transfer functions of the neurons are continuous, it is possible to implement the last model in analog electronic hardware, i.e consisting of non-digital elements. The time involved with parallel, non-digital processing in such an electronic path developer will be several factors smaller than than the movement time of the robot.

The main disadvantage in this study was that the last trajectory formation network had to be simulated on a sequential computer. The large number of calculations in the algorithm caused relatively long processing times. Nevertheless, the wave propagation neural network as a sequential algorithm was fast enough to be used in real-time robotics. In collaboration with the universities of Amsterdam and Utrecht, this model was successfully used in a demonstration project in which the objective was to control movements of a 6 degree-of-freedom robot manipulator by means of neural networks only.

Chapter 2

Population coding in a neural net for trajectory formation

Abstract

In this study we investigate the time-evolution of the activity in a topographically ordered neural network with external input for two types of neurons: one network with binary-valued neurons with a stochastic behavior and one with deterministic neurons with a continuous output. We will demonstrate that for a particular range of lateral interaction strengths, changes in external input give rise to gradual changes in the position of clustered neural activity.

The theoretical results will be illustrated by computer simulations in which we have simulated a neural network model for trajectory planning for a multi-joint manipulator. The model gives a collision free trajectory by combining the sensory information about the position of target and obstacles. The position of the manipulator is uniquely related to the clustered activity of the population of neurons, the population vector. The movement of the manipulator from any initial position to the target position is the result of the intrinsic dynamics of the network.

Adapted from: R. Glasius, A. Komoda and C.C.A.M. Gielen, Population coding in a neural net for trajectory formation, *Network, Computation in Neural Systems*, 5: 549-563, 1994.

2.1 Introduction

Both in neurobiological and artificial (robot) systems, trajectory planning and formation is one of the major objects of study because it is one of the essential aspects in motor control, which is not yet fully understood. The early models for trajectory planning in robot control were based on algorithms which were valid only in some very specific (unnatural) conditions or which required too much time for real-time implementation.

The first models for trajectory planning dealt with static environments only [4, 12, 35, 41, 47, 76] and used global methods, which can be generally viewed as constructing a graph which is used in a search process for an accessible path in a cluttered environment. These models could only be used for static environments because a moving object or introduction of a new object requires that the whole graph has to be constructed anew. Moreover, when multiple obstacles are present, the complexity of the graph increases exponentially which makes it impossible to plan trajectories in real-time.

An other approach, which was proposed later, was the potential field method [36, 42] which improves upon the graph-search in having only linear complexity and non-discrete paths. Unfortunately, the potential field methods suffer from several problems, one being that of undesired local minima, which cause the system to become trapped by some configurations of obstacles [36, 42]. As a consequence, these models can not guarantee a solution to the trajectory-planning problem.

A more successful model is the wave propagation or distance transform model [33]. This approach does not suffer from local minima. If a solution exists the model will find it. However, the path consists of a series of discrete steps on a lattice which is the result of a supervised incremental algorithm to find the next node for the trajectory in the lattice by selecting the neighboring node with the largest activity. In addition to the incremental algorithm additional modules have to be inserted in order to calculate a smooth path along the discrete via-points and in order to reach positions which are not lattice nodes.

Other models have tried to improve upon existing models for robot control related to trajectory formation. For example, Ritter et al. [59, 60] introduced a neural network which was able to learn the transformation from cartesian space coordinates to joint coordinates. In another neural network approach van der Smagt [71, 70] used a backpropagation conjugate gradient neural network to control a robot manipulator which received input about targets from a hand held camera. The neural net learned the transformation from target position in camera domain to a motor command corresponding to the joint angles in

order to move the manipulator in a few straight steps to the target. Both these networks do not specify the trajectory and do not take into account obstacles between the robot and the target position. Therefore, the aim of this study is to develop a model for trajectory planning, which does not have the disadvantages of the previous models for trajectory planning and which is complementary to the inverse kinematics models by Ritter et al. [59] and van der Smagt [71].

In this study we will provide a neural network model, which generates a movement trajectory towards a target position by the intrinsic dynamics of the network, i.e. without a supervisor. The trajectory is a smooth path in configuration space. The model also performs in a cluttered environment in which objects are moving in time.

We will give a theoretical analysis of the dynamics of a topographically ordered neural network, which receives external (for example sensory) input signals, and we will analyze how the evolution of the neural activity can be interpreted in terms of the macroscopic variables. This will be done for two types of networks. The first network is a stochastic system. The activity of neurons is represented as a two-valued random variable and the time evolution of the network is given by the Glauber dynamics [25]. The second network is a deterministic system with activity of neurons given by a continuous real-valued variable. The dynamics of the network is given by a set of non-linear differential equations. In both cases the architecture of the network is the same. The results of the theoretical analyses will be illustrated by computer simulations of trajectory planning and trajectory formation in a topographically ordered map representing a cluttered environment. The system can also be used to control the movement of a multi-jointed manipulator in the presence of obstacles. By obstacles we will understand the regions of space which are occupied by objects or by constraints imposed on joints of the manipulator.

The paper is organized as follows: In section 2 we will define the model of discrete and continuous neurons. We will assume that the activity of each neuron contributes to the position coding of an object or manipulator. The specific position is not coded by an individual neuron but rather by the population of neurons, which we call a cluster of active neurons. The weighted vector sum of the positions coded by these neurons, the population vector, gives the position of the manipulator in space. In section 3 we will describe computer simulations to illustrate the performance of the networks and to confirm our analytical results obtained in 2.

2.2 The models

Both models consist of N neurons arranged in a two-dimensional regular lattice. However, both models can be easily extended to higher dimensions and may have other types of lattices. The neurons have two types of lateral interactions: Short range excitatory and long range inhibitory connections. We assume that the neural activity in the lattice gives a discrete topographically ordered representation of the configuration space $\mathcal{C}$ of a two-joint manipulator. We will call it the neuronal space $\mathcal{N}$. The activity of each neuron i in $\mathcal{N}$ represents events in a particular subset of $\mathcal{C}$, called its receptive field (rf_i). By the configuration space of the manipulator we understand the set of all its possible configurations. The dimension of this space is equal to the minimal number of parameters which are necessary to define any of these configurations uniquely. In this particular case the configuration space is a two-dimensional cartesian space with the two joint-angles of the manipulator along the coordinate axes.

We also assume that external input will provide information about the position of the targets and obstacles in the external world.

The activities of the neurons describe the state of the system. The set $\mathcal{P}$ of all possible states of the network will be called the phase space of the system.

2.2.1 Discrete neurons

In this model we will assume that the N neurons in a topographically ordered map can be in one of the two states

$$\sigma_i \in \{-1, 1\} \qquad i = 1, ..., N \tag{2.1}$$

Since each neuron can be either in the state +1 or -1, the system has 2^N possible configurations. Each of these states represents a corner of the hyper-cube in R^N, so the phase space $\mathcal{P} = \{-1, 1\}^N$.
The input to the neurons is partly provided through lateral connections and partly by external input which provides information about the position of objects in the external world. It is assumed that the external input clamps the activity of all neurons which correspond to obstacles, to the value -1. The lateral symmetric connections consist of short range and long range interactions. The short range interactions B_{ij} between neurons i and j are defined by

$$B_{ij} = \begin{cases} 1 & \text{if } 0 < ||i - j|| < r \\ 0 & \text{otherwise} \end{cases}$$

where r is a positive number and where $||i-j||$ stands for the Euclidean distance between neuron i and j in the lattice. The long range inhibitory connections J_{ij} are given by

$$J_{ij} = \begin{cases} \frac{J_0}{N} & \text{if } i \neq j \ , \ J_0 > 0 \\ 0 & \text{otherwise} \end{cases}$$

The external input to the neuron i contains information about the target-position $\vec{\theta}_T$ and is given by

$$-\alpha \|\vec{\theta}_i - \vec{\theta}_T\| \tag{2.2}$$

where the vector $\vec{\theta}_i$ represents the position of the receptive field rf_i of neuron i in the configuration space and where $\alpha > 0$ is a constant.
The input to each neuron contains also a bias term

$$-J_0 \mu \tag{2.3}$$

This term will appear to be relevant for the number of active neurons in the system.

We define an energy function of the system by the following Hamiltonian.

$$\begin{aligned} \mathcal{H} &= -\sum_{ij} T_{ij}\sigma_i\sigma_j - \sum_i J_0\mu\sigma_i + \sum_i \alpha\|\vec{\theta}_i - \vec{\theta}_T\|\sigma_i \\ &= -\sum_i u_i\sigma_i \end{aligned} \tag{2.4}$$

where $T_{ij} = B_{ij} - J_{ij}$ and where u_i is the total input to neuron i which can be written as

$$\begin{aligned} u_i &= \sum_j B_{ij}\sigma_j - \sum_j \frac{J_0}{N}\sigma_j + J_0\mu - \alpha\|\vec{\theta}_i - \vec{\theta}_T\| \\ &= \sum_j B_{ij}\sigma_j - \sum_j \frac{J_0}{N}\sigma_j + J_0\mu - \alpha x_i \end{aligned} \tag{2.5}$$

Here we use the vector $\vec{x}_i$ of $\vec{\theta}_i$ relative to $\vec{\theta}_T$ defined by

$$\vec{x}_i\left(\vec{\theta}_T\right) \equiv \vec{\theta}_i - \vec{\theta}_T \quad \text{and} \quad x_i \equiv \|\vec{x}_i\| \tag{2.6}$$

Each component of the Hamiltonian has a different effect on the behavior of the network. The long range inhibitory interactions J_{ij} contribute to the stabilization of the total amount of neural activity in the network near the value μ. The local neighbor interactions B_{ij} make it more advantageous for neighboring

units to be in the same state. Therefore, this term tends to impose clustering of the units having the same activity. Finally the external input $-\alpha x_i$ tends to shift the activity towards the target position $x_i = 0$. The relative magnitudes of these three terms determine the properties of the system.

The dynamics of the system is sequential and stochastic: at each time step only one neuron, chosen at random among the N neurons, is updated. The transition probability w_i, that neuron i changes its activity from σ_i to -σ_i while the others remain momentarily fixed, is

$$w_i(\vec{\sigma}) = \frac{1}{2}\left[1 - \tanh\left(\beta u_i \sigma_i\right)\right] \tag{2.7}$$

where $\beta = \frac{1}{T}$ is the inverse of a noise-parameter T. In the zero noise limit ($T = 0$), the probabilities w_i become 0 or 1 and the updates become deterministic

$$\sigma_i(t+1) = \text{sign}\left(u_i(t)\right) \tag{2.8}$$

However, even for the case $T = 0$ the time evolution of the system is still stochastic due to the random choice of a neuron at each time step.

To give these microscopic states a physical interpretation we will define some macroscopic variables. The first macroscopic variable is called the magnetization m

$$m \equiv \frac{1}{N}\sum_i \sigma_i \tag{2.9}$$

and is related to the total activity in the whole population of neurons. As a result the total input can be rewritten as

$$u_i = \sum_j B_{ij}\sigma_j - J_0\left(m - \mu\right) - \alpha x_i \tag{2.10}$$

The second macroscopic variable of interest is a vector which codes the actual position of the manipulator relative to the target position in configuration space.

$$\vec{X}_B \equiv \vec{\theta}_B - \vec{\theta}_T \tag{2.11}$$

where $\vec{\theta}_B$ is the actual position of the manipulator in configuration space

$$\vec{\theta}_B \equiv \frac{\sum_i \vec{\theta}_i\left(\sigma_i(t)+1\right)}{\sum_i\left(\sigma_i(t)+1\right)} \tag{2.12}$$

$\vec{\theta}_B$ corresponds to the center of gravity of the activities in the neural map. The position vector $\vec{X}_B$ which is a weighted vector sum of the contributions of all

neurons, provides information about the position in configuration space. The k-th component of the vector $\vec{X}_B$ is defined by

$$X_B^k(t) \equiv \frac{\sum_i x_i^k(\sigma_i(t)+1)}{\sum_i(\sigma_i(t)+1)} \tag{2.13}$$

The length of the vector $\vec{X}_B$ is the distance in configuration space between the actual configuration and target configuration.

2.2.2 Continuous neurons

In this model the N neurons in the topographical map may assume a real output value in the interval $[-1,1]$

$$s_i \in [-1,1] \qquad i = 1,...,N. \tag{2.14}$$

Hence in this model $\mathcal{P} = [-1,1]^N$, where $[-1,1]^N$ stands for a hyper-cube in R^N. The input to the neurons is similar to the case of the discrete model. The total input of neuron i can be written by

$$u_i = \sum_j B_{ij}s_j - J_0(m-\mu) - \alpha x_i \equiv \sum_j^N T_{ij}s_j(t) + I_i \tag{2.15}$$

with

$$I_i \equiv J_0\mu - \alpha x_i \tag{2.16}$$

The activity of neuron i is given by the sigmoid function g over the total input:

$$s_i = g(u_i) \tag{2.17}$$

The system dynamics, in this case, is parallel and deterministic

$$\frac{du_i(t)}{dt} = \sum_j^N T_{ij}\, s_j(t) + I_i - u_i(t) \tag{2.18}$$

It has been shown before [10, 24, 31] that the function

$$L(\vec{s}) = -\frac{1}{2}\sum_{i,j} T_{ij}\, s_i\, s_j - \sum_i I_i s_i + \sum_i G(s_i) \tag{2.19}$$

with $G(s_i) = \int_0^{s_i} g^{-1}(x)\,dx$, is a global Liapunov function for (2.18) with the sigmoid function $g(x) = \tanh(\beta x)$ and $\beta > 0$. The vector $\vec{s}$ refers to the state of the neural network with components s_i. The existence of a Liapunov function

guarantees that the dynamics of the network always converges to a fixed point which is an equilibrium state of the system. By direct computation,

$$\frac{dL(\vec{s}(t))}{dt} = \sum_i^N \frac{\partial L(\vec{s}(t))}{\partial s_i} \frac{ds_i(t)}{dt} = -\sum_i^N \left(\frac{du_i(t)}{dt}\right)^2 g'(u_i) \leq 0$$

$g'(.)$ refers to the first derivative of the function $g(.)$ with respect to its argument. The global stability of the system defined by (2.18) and (2.19) is the same as that studied by Cohen [10], Grossberg [27] and Hopfield [31].

The population vector $\vec{X}_B$ which gives the relative actual position, is now defined as

$$X_B^k(t) \equiv \frac{\sum_i x_i^k (1 + \text{sign}(s_i(t)))\, s_i(t)}{\sum_i (1 + \text{sign}(s_i(t)))\, s_i(t)} \tag{2.20}$$

Since the population vector is a function of time, it can be seen as a neuronal representation for the position in time relative to the target position. Each pattern of neural activity corresponds to a position in space via (2.20) and the time evolution of $\vec{X}_B$ is described by the time evolution of the neural activity.

Due to external input, which provides information about the target position, the activity of neurons begins to change according to the dynamics of the network described by equation (2.18). This evolution can be seen in the phase space $\mathcal{P}$ as a motion of a point along the curve on which the Liapunov function decreases. This down-hill motion ends when the network reaches an equilibrium state. The interpretation of the population vector transforms this curve from the phase space into a curve in the configuration space.

The equation (2.19) represents a free energy F of the discrete system (2.4) in the mean field approximation [56]:

$$\begin{aligned} F_{\text{MFA}} &= -\frac{1}{2} \sum_{i=1,j=1}^N T_{ij} s_i s_j - \sum_{i=1}^N I_i s_i + \\ &\quad \frac{1}{2\beta} \sum_{i=1}^N ((1+s_i)\, ln\,(1+s_i) + (1-s_i)\, ln\,(1-s_i)) \end{aligned} \tag{2.21}$$

and is an upper bound of the exact free energy of the system. The last term in (2.21) is the entropy of the system and is exactly equal to the last term in the Liapunov function (2.19). To obtain the equilibrium free energy and the best upper bound on F one has to minimize the right hand side of (2.21) with respect to all u_i. This minimization procedure can be seen as a dynamical process given by (2.18).

2.3 The simulations

To illustrate the properties of the two models we simulated a two-dimensional neural network with $N = 2500$ neurons, arranged in a 50x50 grid. The neurons are placed on the nodes of a square lattice. However, the model can be easily extended to higher dimensions and may have other types of lattices. The neural map is a discrete representation of the two-dimensional configuration space. The configuration space variables $\theta_1, \theta_2 \in [0, 100]$. Hence each neuron has a square shaped receptive field with sides of $\gamma = 2$.

By choosing $r = 1.5$ each neuron on the lattice is connected with excitatory connections of strength 1 to its $D = 8$ neighbors. The neurons also interact with each other through long range inhibitory connections of strength J_0.

Each initial state of the simulations consists of a small connected group of neurons with activity 1. The center of mass of their activity corresponds to the initial position of the manipulator in configuration space. All other neurons have an initial value of activity -1.

Information about the position of the target and obstacles are supplied to the network by external inputs. The input to each neuron i due to the target is αx_i. The obstacles are represented in the neuronal map by clamping the activity of the neurons which correspond to the obstacle positions, to the value -1.

2.3.1 Discrete neurons

The dynamics of this model is the stochastic sequential heat bath: at each time step a neuron i is selected at random, and subsequently u_i was calculated using equation (2.10). Finally, this neuron was updated according to equation (2.7).

The total input to the network is given by equation (2.10). Denoting the actual number of neurons in state "1" by n^+ and the desired number of neurons in state "1" by $\tilde{n}^+$, the magnetization m and the desired magnetization μ can be expressed respectively as

$$m \equiv \frac{2n^+}{N} - 1 \quad \text{and} \quad \mu \equiv \frac{2\tilde{n}^+}{N} - 1 \tag{2.22}$$

Then the total input for neuron i can be rewritten to

$$u_i = \sum_j B_{ij}\sigma_j - \frac{2J_0}{N}\Delta n - \alpha x_i \tag{2.23}$$

with $\Delta n \equiv (n^+ - \tilde{n}^+)$ is the difference between the actual and the desired number of neurons in state "1".

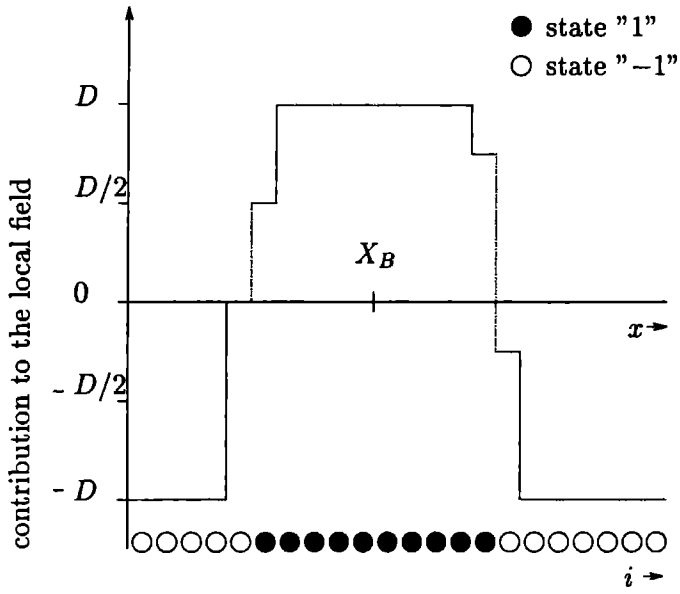


Figure 2.1: The 'short range interaction' contribution to the local field given for the neurons i along a one-dimensional cross section of the neuronal map.

Equation (2.23) for the total field of neuron i contains three terms.

The 'short range interaction' term $\sum_j B_{ij}\sigma_j$ causes clustering of active neurons. If all D neighbors of a neuron are active, then the 'short range interaction' contribution to that neuron is D. If all D neighbors are non-active the contribution is $-D$. The neurons at the boundaries, lying between positive- and negative-regions, receive a contribution to their local field by a value between $-D$ and D. The short range interaction contribution to the local field of the neurons located on a line passing through the center of the cluster, is shown in figure 2.1.

The 'magnetization' term $-J_0\,(m-\mu)$, which has been rewritten to $-\frac{2J_0}{N}\Delta n$, facilitates the stabilization of the number of neurons in state "1". For each neuron in state "1" which changes to state "-1", the local field of all neurons is raised by an amount of $\frac{2J_0}{N}$.

The 'dragging' term $-\alpha x_i$ is the driving force of the cluster towards the target. The larger the distance between the receptive field of a neuron and the target, the more the neuron will be inhibited. Therefore, neurons with a small relative

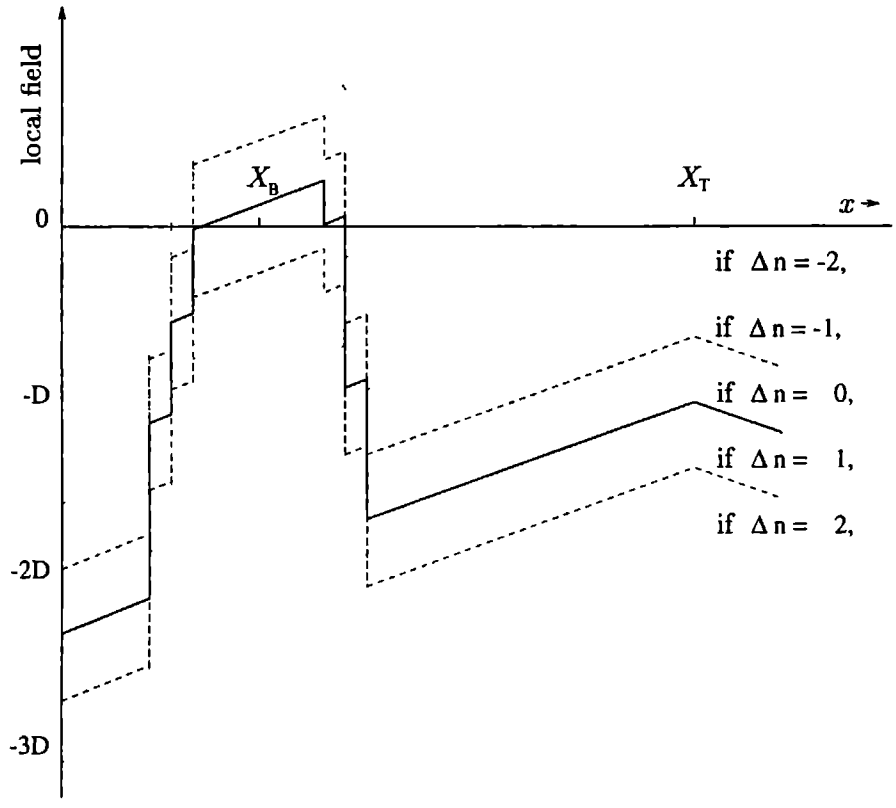


Figure 2.2: The total local field, the result of the weighted contribution of the three input terms, as a function of the relative distance x along the line between actual position and target position for different numbers of active neurons.

distance towards the target have higher local fields.

To see the interplay of the three terms we have to examine the total local field (2.23). Each of the curves in figure 2.2 shows the contributions of the three terms to the local field. The absolute value of the slope of the curves is equal to α. The vertical position of the curve on the diagram depends on the 'magnetization' term: The more positive this term is, the lower the position of the curve.

The vertical distance between various curves corresponds to different numbers of active neurons, each giving rise to a difference in the local field by $\frac{2J_0}{N}$. Each curve has two maxima, one at the target position and one at the cluster position.

It depends upon the relative distance between cluster and target which maximum is larger.

Because the behavior of the system depends on the **relative** magnitudes of the three parameters B_{ij}, J_0 and α and since we have already fixed $B_{ij} = 1$, the only free independent parameters are J_0 and α.

We want to chose values for J_0 and α such that the number of active neurons, during the time-evolution of the system, is close to $\bar{n}^+$ and that they form a connected cluster. This allows us to interpret the activity of the population of neurons in terms of macroscopic variables defined in equation (2.13).

To estimate J_0 and α we will reconsider the energy function (2.4) in a new form

$$\mathcal{H} = -\frac{1}{2}\sum_{ij} B_{ij}\sigma_i\sigma_j - \frac{NJ_0}{2}\left(m^2 - 2m\mu\right) + \sum_i \alpha x_i\sigma_i \tag{2.24}$$

where we used (2.9). The first term in the energy function is at most of order ND and has its minimum when all neurons are in the same state. The last neuron is at most of order $\alpha N^{\frac{2}{3}}$ and forces all neurons to have activity "-1". The middle term has its minimum at $m = \mu$ and is then equal to $-\frac{1}{2}J_0N\mu^2$. Since the number N of neurons in the network is large but finite, the contribution from the first and third term will cause that m is not precisely equal μ. To construct a system with magnetization m as close to the desired magnetization μ as possible, the middle term must be the dominating term. If, for example, we require that $(m - \mu) \propto N^{-1}$ then from (2.24) it follows that $J_0 \propto N$ and $J_0 \approx \alpha N^{2/3}$. This gives $\alpha \propto N^{-1/2}$.

The initial configuration of the system consists of a closed, circular cluster of activity, placed some distance away from the target position. The size of the initial cluster is always such that $m = \mu$. With these settings, the target neuron has a local field of $-D$.

If $\alpha \geq D$ then all neurons, at time zero, have negative local fields. Because changes in the state of neurons occur only if the local field and the state of the neuron have opposite signs, only active neurons can change their sign. If one active neuron changes its state, then $\Delta n = 1$ and the local fields of all neurons are incremented by an amount of $\frac{2J_0}{N}$. If still all neurons have negative fields, another neuron in the cluster changes state to "-1" and again all local fields are raised by $\frac{2J_0}{N}$. This continues until one of three possible situations occurs:

A) No active neurons are left to change their state to "-1" and all neurons still have negative local fields. This happens when $\alpha \geq D + J_0(\mu + 1)$. This is a stable state of the system. All neurons which were initially in state "1", have

changed their state to the rest value "-1" and the local field of all neurons is raised by $\frac{2\tilde{n}+J_0}{N}$. However, the elevation was not large enough to raise one of the maxima above the zero level.

B) The local field of the neurons in the cluster exceeds that of the neurons in the target and is raised by successive steps to a positive value. This will happen when J_0 and α satisfy the following inequalities.

$$J_0 \leq \frac{D}{(m-\mu)} \quad \text{and} \quad \alpha \leq \frac{D}{(m-\mu)\,N^{\frac{2}{3}}} \tag{2.25}$$

Then neurons in the cluster, having all D neighbors active, have positive local field. Neurons outside the cluster, having negative valued neighbors only, have negative local fields. The instable neurons, having opposite state and local field, are those located near the border of the cluster. These neurons do not have all their neighbors in the same state.

As a result of the changing 'magnetization term' these border neurons are responsible for the movement of the cluster. The border neurons at the side of the cluster directed towards the target, have higher local fields relative to the border neurons at the opposite side of the cluster. In the case that Δn is negative, the system tends to change the state of a border neuron from "-1" to "1". This will happen with larger probability at the target side of the cluster. In the same way, if Δn is positive and the system tends to turn the state of a neuron from "1" to "-1", this will happen with larger probability at the opposite side of the cluster. Both processes result in a shift of the cluster towards the target representation in the map.

C) The local field of the target neurons exceeds that of the neurons in the cluster and the local field of these neurons is raised by successive steps to a positive value. The cluster neurons having negative local fields change their state from "1" to "-1". The target neurons having positive local fields change their states from "-1" to "1". Hence the cluster disappears at its initial position and appears at the target position. The cluster seems to jump.

To increase the speed of movement of the cluster, we can increase α during the movement towards the target while J_0 is fixed. Because the distance of the cluster relative to the target representation decreases, the 'dragging' term for the cluster neurons becomes smaller. Consequently the influence of the 'magnetization' term becomes more noticeable which results into a smaller difference between m and μ. Regarding equation (2.25) this gives us the opportunity to increase α. A larger α results in a stronger 'dragging' field and a higher velocity of the movement of the cluster.

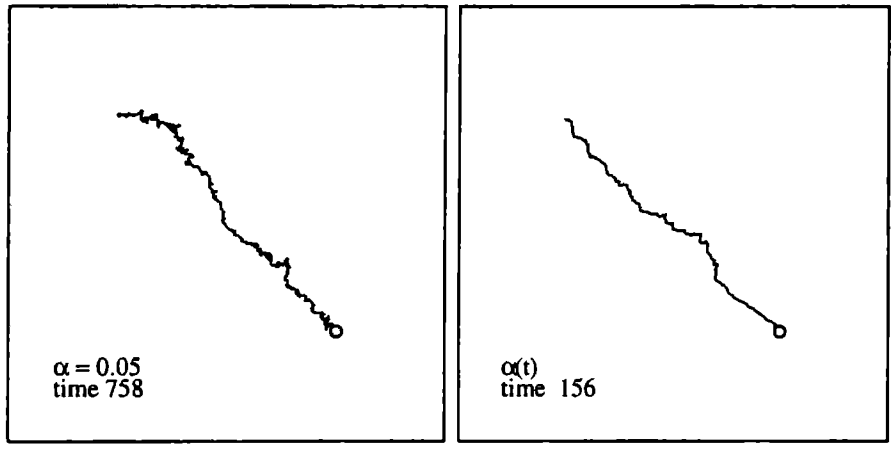


Figure 2.3: (a) A trajectory of the center of the cluster with active neurons as the result of the dynamics of the neural network with α constant. The position of the target is shown by the circle. (b) The simulation with optimal $\alpha(t)$. Both simulations were done with parameters $N = 2500$, $\tilde{n}^+ = 50$, $J_0 = 2500$ and $\beta = 2$. Note that the duration of the movement towards the target is longer in A than in B.

We performed simulations with constant α and time-varying $\alpha(t)$, with and without obstacles. Initially we chose $\alpha = 0.05$ and $\mu = -0.96$ which corresponds to a desired number of $\tilde{n}^+ = 50$ active neurons. We chose $J_0 = 2500$.

Figure 2.3(a) shows a typical result of a simulation with constant $\alpha = 0.05$. Figure 2.3(b) shows a result for the optimal value of $\alpha(t)$. Obstacles were not included.

In both cases the cluster moved in small discrete steps as a result of the dynamics of the neurons. The successive positions of the center of mass of the cluster present the trajectory in the configuration space. Because of the stochastic character of the model each simulation resulted into a different path.

In the case that $\alpha = 1.0$ was chosen too large, the cluster disappeared at the initial position and appeared, due to the 'magnetization' term, at the position with the highest local field. The cluster jumped to the target.

Figure 2.4 shows the typical trajectory of the cluster in the presence of obstacles for optimal $\alpha(t)$. In the presence of noise ($T > 0$) and because of the random

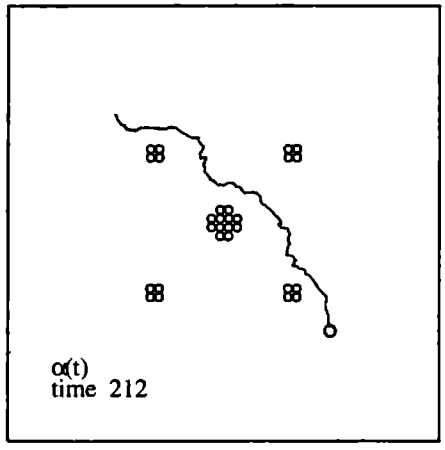


Figure 2.4: A trajectory of the center of the cluster with active neurons in the presence of obstacles for optimal $\alpha(t)$. The position of an obstacle is shown by the circles. The simulation was done with parameters $N = 2500$, $\tilde{n}^+ = 50$, $J_0 = 2500$ and $\beta = 2$.

choice of a neuron at each time step, with $\alpha < \alpha(t)$ the jump of the cluster to the target position may still occur. The probability of a jump however, decreased with decreasing noise. For $\beta = 5$ these jumps were rarely seen.

The motion of the cluster can be seen as the Brownian motion in the presence of an external deterministic field αx_i. For the large parameter $\alpha(t)$ the deterministic dragging part becomes more important relative to the stochastic part. Hence the fluctuations decreased, the trajectory became more straight and the time to reach the target decreased.

2.3.2 Continuous neurons

In the simulations with continuous neurons we used a discrete-time version of the dynamics (2.18). At each time step Δt all neurons changed their state according to

$$u_i(t + \Delta t) = u_i(t) + \Delta t \left(\sum_j^N T_{ij}\, g(u_j(t)) + I_i - u_i(t) \right) \tag{2.26}$$

The sigmoid function $g(x) = \tanh(\beta x)$ where β was the gain parameter.

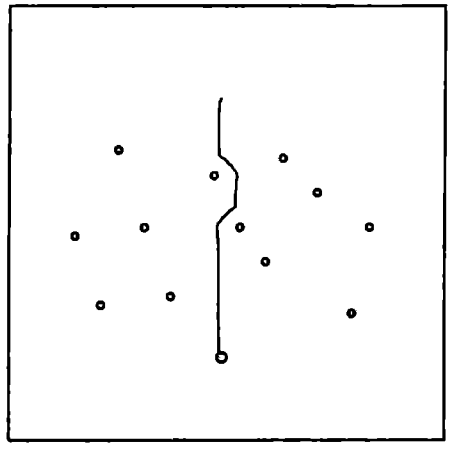


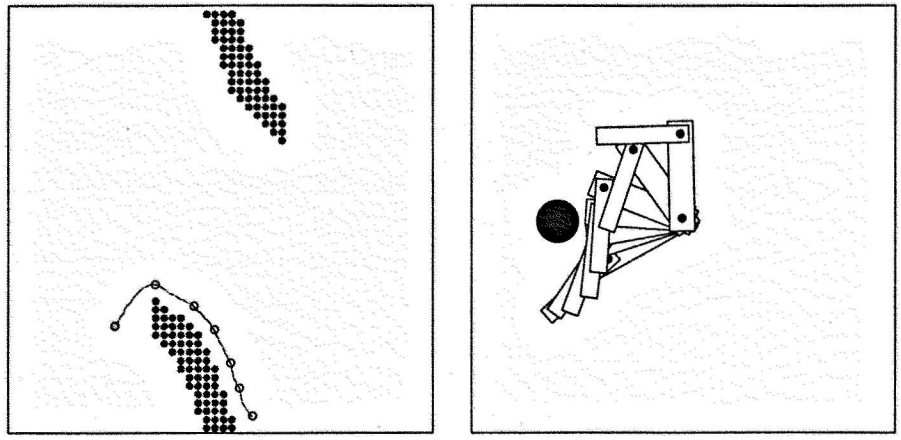
Figure 2.5: A continuous trajectory as a result of the dynamics of continuous neurons in the presence of obstacles. The continuous line is the movement of the center of activities of the cluster in time. The gray circle corresponds to the target configuration and the obstacle positions are represented by the open circles.

We chose $\beta = 0.1$, $J_0 = 2500$, $\alpha = 0.05$ and $\mu = 0.988$ which corresponded to a cluster with fifteen active neurons. Because of the deterministic nature of this network, the paths which are generated by the network, are the same for all values of $\alpha < \alpha(t)$. The value of α only affects the speed of the movement of the cluster. Created trajectories were straight lines except for the case when the presence of obstacles prevented the straight line.

The results are illustrated in figure 2.5. Since the function (2.19) decreased on the trajectories given by (2.18) and was bounded from below, the system reached an equilibrium state in which the cluster reached the position of the target.

2.3.3 A Two-link Robot Manipulator

In this simulation we use the continuous model to control a two-link robot manipulator in the vicinity of an obstacle. The configuration space is a set of all possible configurations of the manipulator. All configurations with an overlap of manipulator and obstacle are called obstacle configurations. The neural map has the topographical order of the configuration space and is a discrete representation



(a) (b)

Figure 2.6: (a) Configuration space of the manipulator. The dark dots represent the positions of the obstacle neurons. The smooth line is the path generated by the model. The open circles indicate intermediate positions of the cluster in configuration space corresponding to the manipulator configurations shown in figure: (b)) World space. Some intermediate manipulator configurations during the movement from the initial to the target configuration. The dark circle is the obstacle in world space.

of it.

Figure 2.6(a) shows the configuration space with the positions of the obstacle neurons and the path as a result of the system dynamics. The open circles on the path correspond to the configurations of the manipulator in world space shown in figure 2.6(b).

Note that without the obstacle the path in configuration space, between the initial and target position, would be a straight line. In the precense of the obstacle the trajectory follows a curved path in order to avoid a collision with the obstacle.

Figure 2.7 shows the local field (left column) and the neural activity (right column) of the neurons in the network at three moments of time during the trajectory formation. The trajectory is given by the movement of the activity cluster in the neural map. The obstacles are represented by neurons with a negative local field.

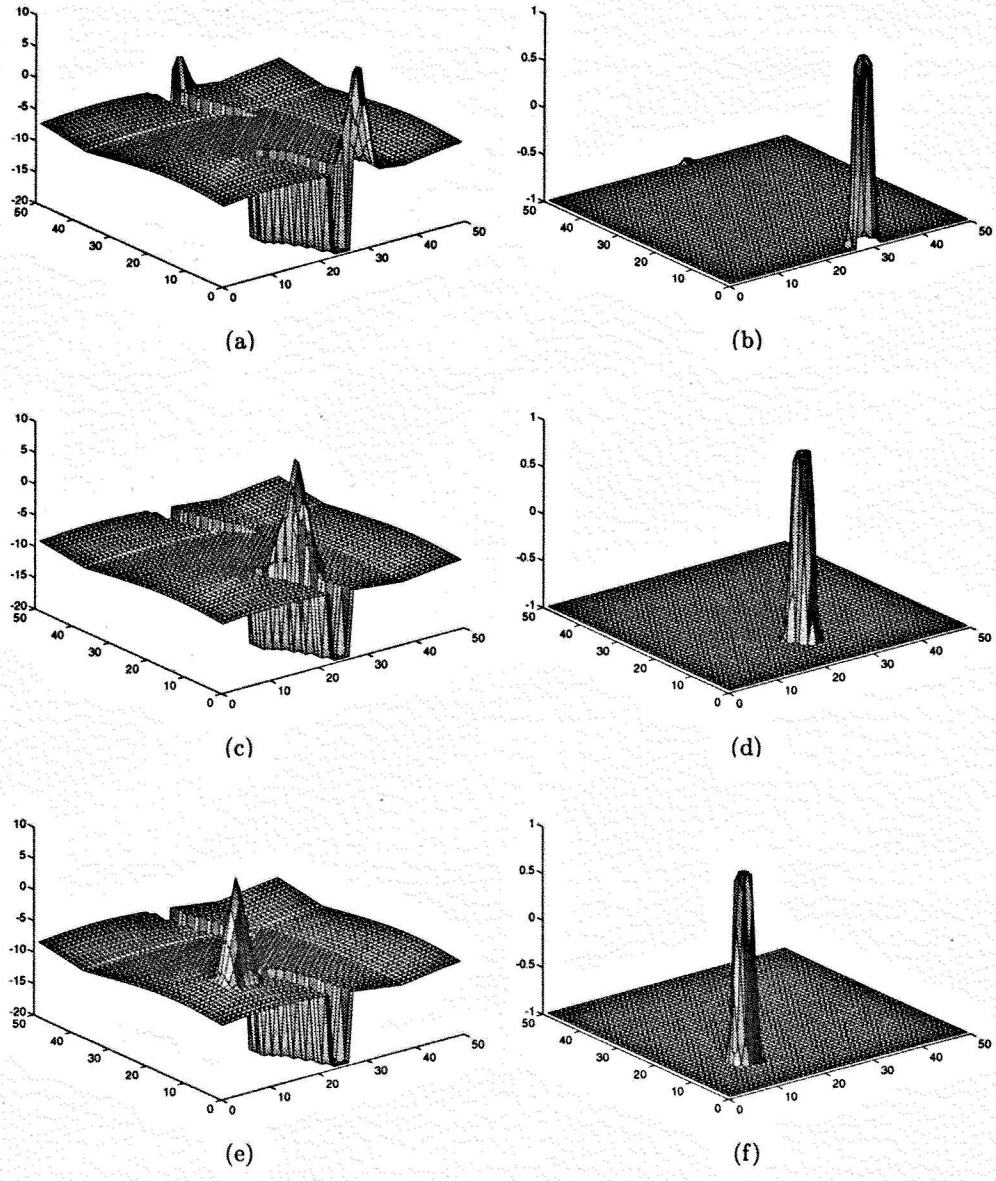


Figure 2.7: Local field (left column) and neural activity (right column) at three moments of time (at the beginning ((a),(b)), at the end ((e),(f)) and at an intermediate time ((c),(d)).

2.4 Discussion

In this paper we presented a neural network for trajectory planning and formation. The network consists of a large number of neurons connected by two types of interactions: short range excitatory and long range inhibitory connections. All neurons with positive activity formed a connected cluster due to the short range interactions. The total activity of these neurons was kept constant (neglecting small fluctuations in the case of discrete neurons) due to long range inhibitory connections. When the network received information about the position of the target, the initial activity of neurons started to change. During the time evolution the cluster moved to the target and the total activity of the network was constant. To give a macroscopic interpretation of the activity of the population of neurons we defined a vector-valued function. This function mapped the state of the network, the pattern of activity of neurons, into the position of the manipulator in configuration space. Encoding information about the position of the manipulator by the population of neurons gives the possibility to generate a smooth motion of the manipulator.

With a cluster consisting of many active neurons, the position of the manipulator corresponds to the position of the center of gravity of all active neurons. Therefore, in the model with discrete neurons, the trajectory of the manipulator through the configuration space has a resolution better than the resolution of the configuration space corresponding to the discrete grid of neurons. In the continuous neuron model and with continuous time, the path is also continuous. The network could also control the movement of the manipulator in the presence of obstacles. However, the performance of the network was limited only to obstacles of very regular and convex shape, because the dragging term $-\alpha x$ might drive the system into a local minimum for non-convex shaped objects. If obstacle and target move then the local minima change or may even disappear.

We studied a network with discrete and graded neurons. In both cases the network could control the movement of the manipulator from any initial configuration to any final configuration. However, there are a few properties which are worth mentioning. In the case of discrete neurons the system is stochastic. The movement of the activity cluster is similar to the Brownian motion of a particle in the presence of a deterministic external field. In the case of graded neurons the system is deterministic. This feature gives the possibility to make the system much faster than the previous one by parallel hardware implementation. The size of the cluster can change from a few neurons to hundreds of neurons and the system can still perform correctly.

Just like the potential-field methods, our model can be trapped in local min-

ima. However, it is preferable over potential-field methods for the following reason. If the potential-field method is implemented in hardware for fast real-time applications, the accuracy of the method depends on the resolution of the discrete grid which represents the configuration space. For real-time applications our neural network is also implemented on a grid, but the resolution is much higher than that of the discrete potential field because the relative actual position is the weighted average of the cluster of activity. Our method has continuous dynamics and generates a continuous trajectory, contrary to the discrete potential field.

The continuous model possesses some biological features. For example, it has been shown in neurophysiological experiments that information is not coded by a single neuron, but rather by a group of active neurons, which are frequently clustered together. An example may be the superior colliculus which plays an important role in the generation of saccadic eye movements [52]. The neurons in the deep layers of the superior colliculus have a movement field, which is defined as the set of vectorial eye movements for which the neuron will be active. The neurons are organized in a topographically ordered way in the sense that neighboring neurons have neighboring movement fields [69]. Recent experiments [52] have shown that at the beginning of a saccade a particular cluster of neurons is active. This cluster of active neurons shifts along the superior colliculus map until the eye is on target. For these observations a model has been proposed [14], which basically represents a shift of a cluster of active neurons in a topographical map. The present paper gives a more mathematically elaborated version of that model and in addition explains how the "shifting" cluster can avoid obstacles.

In the last few years the notion of "population vector" was used frequently for interpreting neuronal responses in motor cortex. The activity of neurons in motor cortex is related to arm movements [9, 18, 34]. Because it not known whether moto-cortical neurons are organized in a topographically ordered way, it is not possible yet to apply the model directly to the dynamics of the shifting population vector. However, the model helps to understand the characteristics of the population vector.

Summarizing, we have presented a neural network, which by its intrinsic dynamics can generate a continuous trajectory without collisions with obstacles. This is true even when the obstacles are moving.

Acknowledgments

We would like to thank Ton Coolen for suggesting this approach.

Chapter 3

Neural network dynamics for path planning and obstacle avoidance

Abstract

A model of a topologically organized neural network of a Hopfield type with nonlinear analog neurons is shown to be very effective for path planning and obstacle avoidance. This deterministic system can rapidly provide a proper path, from any arbitrary start position to any target position, avoiding both static and moving obstacles of arbitrary shape. The model assumes that an (external) input activates a target neuron, corresponding to the target position, and specifies obstacles in the topologically ordered neural map. The path follows from the neural network dynamics and the neural activity gradient in the topologically ordered map. The analytical results are supported by computer simulations to illustrate the performance of the network.

Adapted from: R. Glasius, A. Komoda and C.C.A.M. Gielen, Neural network dynamics for trajectory formation and obstacle avoidance, *Neural Networks*, 8: 125-133, 1995.

3.1 Introduction

Human motor control reveals a versatility of function and economy of space, that is yet beyond the reach of robots. One of the important themes of research in the field of robotics is the design of a motion planning system. Such a system should guide a robot or a robot manipulator from an initial position to a target position, avoiding obstacles which are located between these positions. If we take into account that the obstacles as well as the target can move and that the obstacles can have any shape, it becomes clear that this problem is not a trivial one. Especially planning a path for a robot in an environment which is unknown and changing, is a difficult problem.

In the past, several authors [4, 12, 35, 41, 47, 76] have worked on the path planning problem. Most work so far deals with static environments and used global methods, which can generally be viewed as a search process for a path in a graph. However, global methods will limit the real-time capabilities of robots in a cluttered environment, especially when new information about changes in the environment is becoming available continuously, because of the time needed to perform the planning task. Later, the idea of using an artificial potential field around each obstacle was proposed combined with an attractive potential around the target [36, 42]. Unfortunately, the potential functions proposed suffered from several problems, one being that of undesired local minima. Later, several papers have attempted to provide solutions to this problem (e.g. [3, 5, 11, 53, 75]). None of these papers mentioned above offers an exact potential-function-based algorithm that is guaranteed to work. Recently, a new algorithm was proposed which addressed the path-finding problem using an iterative approach under the constraints of minimum time or minimum energy [67]. However, this method will be very time-consuming, especially for many, complex objects in the environment.

In this paper we present a path planning system which can control robot motion in a static as well as in a dynamic environment with a guaranteed solution, which is a shortest path in terms of the metric of the representation, if one exists. We will assume that information about the position and shape of obstacles in the environment is not known beforehand and that it appears during the path planning, for example, through real-time sensing. This may be rather artificial for path planning in a static environment, but it is an inevitable situation for path planning in an environment in which the position of target and obstacles are changing continuously. This makes our approach distinct from other approaches [13, 47, 66], which presume an algebraic representation of a robot environment and which call for one-time off-line computation. In our approach we have only studied path-planning, and not trajectory formation such as in [74].

In the latter approach one has to deal with forward and backward models of the system under control before trajectories can be generated from an initial to a final point. For complex paths, such as in a cluttered or changing environment, the total path has to be segmented in order to create a sequence of trajectories.

Our path planning system is based on a Hopfield-type of network with continuous neurons [30]. Analog neurons provide the possibility to make such a system a fast computing device, which can process the massive amount of sensory data necessary to move in a changing environment. In early attempts to understand biological computation, neurons were modeled as two-state threshold devices only. However, parallel computation with discrete neurons faces stability problems not found in sequential dynamics. However, sequential dynamics - when neuron states are updated one at a time - implemented in software on a conventional computer is simply too slow for any large network application. These stability problems associated with parallel dynamics can be eliminated by using continuous neurons [48]. They also provide the possibility for hardware implementation

The paper is organized as follows: In section 2 we define the model and show its basic properties. We define two types of dynamics: parallel discrete- and continuous-time dynamics. We show for both types of dynamics that the network, after receiving an external stimulus, evolves towards a specific state of neural activity which corresponds to a minimum of a Liapunov function. The path is given by the neural activity gradient. This path has the shortest length among all paths connecting initial and goal position. Computer simulations illustrating the performance of the network are presented in section 3. Section 4 summarizes the conclusions.

3.2 The model

3.2.1 Work space and configuration space

Consider a robot manipulator $\mathcal{R}$, equipped with sensors and actuators, is operating in a subset of the real world. This robot should have the capability of planning its own motion by generating a path specified by an initial configuration, a final configuration and by the position of obstacles in the workspace. We assume that the non-redundant robot manipulator has d degrees of freedom which may correspond to the d joints of the robot arm. In this paper we will only deal with the kinematics of the movement-trajectory of the manipulator. We will assume that the d degrees of freedom span a d-dimensional configuration space $\mathcal{C}$. Each point in $\mathcal{C}$ defines a unique configuration of the non-redundant robot-manipulator in

workspace.

The obstacles $\mathcal{O}_\alpha$, $\alpha = 1 \ldots m$ in workspace , are represented in the configuration space as the subset of those points in configuration space, which cannot be reached by the manipulator. This includes also configurations which are not allowed since it would require that one of the links touches or passes through an object. Therefore, the "forbidden" regions in configuration space are larger than the regions that correspond to the positions of obstacles. In mathematical terms the obstacle regions are defined in $\mathcal{C}$ by the points where the objects and the links of the robot manipulator share at least one point:

$$\hat{\mathcal{O}}_\alpha = \{\mathbf{q} \in \mathcal{C} \mid \mathcal{R}(\mathbf{q}) \cap \mathcal{O}_\alpha \neq \emptyset\}, \tag{3.1}$$

where $\mathcal{R}(\mathbf{q})$ is an area occupied by $\mathcal{R}$ at configuration $\mathbf{q}$ in the workspace.

Feasible paths between an initial configuration $\mathbf{q}_{\text{init}}$ and a target configuration $\mathbf{q}_{\text{targ}}$ are continuous maps from a closed real interval into the free configuration $\mathcal{C}_{\text{free}}$ space which remains after removing the points $\hat{\mathcal{O}}_\alpha$ from $\mathcal{C}$

$$\mathcal{C}_{\text{free}} = \mathcal{C} \setminus \bigcup_{\alpha=1}^{m} \hat{\mathcal{O}}_\alpha \tag{3.2}$$

3.2.2 Neuronal space and phase space

Our path planning system consists of a large collection of identical processing units called neurons. These neurons are arranged in a d-dimensional cubic lattice (CL). The number of neurons in the network is N. They are connected only to their z nearest neighbors by excitatory and symmetric connections T_{ij}. Generalization to other types of lattices can be obtained very easily. We assume that the lattice represents a topologically ordered map, which can be obtained by a Kohonen-type of learning [17, 49, 59]. This map gives a discrete topologically ordered representation of the robot configuration space $\mathcal{C}$ [28, 51, 73] with nodes homogeneously distributed over the configuration space. We will call it the neuronal space $\mathcal{N}$. To each neuron in $\mathcal{N}$ corresponds a certain subset of $\mathcal{C}$ called its receptive field. In this discrete representation of the configuration space $\mathcal{C}$ the regions $\bigcup_{\alpha=1}^{m} \hat{\mathcal{O}}_\alpha$ are represented by a subset of nodes of the CL. We will call the nodes, which correspond to $\bigcup_{\alpha=1}^{m} \hat{\mathcal{O}}_\alpha$, occupied nodes. Similarly the initial configuration $\mathbf{q}_{\text{init}}$ and the target configuration $\mathbf{q}_{targ}$ are represented by one node each. The external input signals, which provide information about the position of obstacles in workspace, are supposed to clamp the activity of all neurons in the occupied nodes to the value "zero" corresponding to the minimal activity of the neuron. The activity of the neuron corresponding to the target configuration

node is clamped to the value "one". The initial configuration of the robot may correspond to any node which is not occupied.

The activities of the neurons are characterized by real-valued variables $\sigma_i \in [0,1]$ and describe the state of the system. The set $\mathcal{S}$ of all possible states σ of the network is called the phase space of the system. In our model $\mathcal{S} = [0,1]^N$, where $[0,1]^N$ stands for a hypercube in $\boldsymbol{R}^N$.

3.2.3 Dynamics

The state variables σ_i, $i = 1 \ldots N$, of the neurons can change due to inputs from other neurons in the network and due to external sensory input. The total input u_i to the neuron i is a weighted sum of activities from other neurons and of an external sensory input I:

$$u_i(t) = \sum_j^N T_{ij}\sigma_j(t) + I_i \tag{3.3}$$

where the strength of the synaptic connection from node j to node i is represented by T_{ij}. In our model the connections T_{ij} are excitatory ($T_{ij} > 0$), symmetric ($T_{ij} = T_{ji}$) and short range:

$$T_{ij} = \begin{cases} 1 & \text{if } \rho(i,j) < r \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

Here r is a positive number and $\rho(i,j)$ is the Euclidean distance between neuron i and j in $\mathcal{N}$.

The evolution of the configuration of this network can be defined by the discrete-time or continuous-time dynamics. In the case of a time-discrete evolution the states of neurons are updated by

$$\sigma_i(t+1) = g\left(\sum_j^N T_{ij}\,\sigma_j(t) + I_i\right) \tag{3.5}$$

where $g(x)$ is a sigmoid function.

The dynamics (3.5) may be parallel, sequential or may be a random , asynchronous sequence. In the continuous-time updating, all neurons continuously and simultaneously change their states. The change of activity of the neurons is then given by the set of nonlinear differential equations [27]

$$\frac{d\sigma_i(t)}{dt} = g\left(\sum_j^N T_{ij}\,\sigma_j(t) + I_i(t)\right) - \sigma_i(t) \tag{3.6}$$

Both types of dynamics lead to the same equilibrium state. The function

$$L(\sigma) = -\frac{1}{2}\sum_{i,j} T_{ij}\,\sigma_i\,\sigma_j - \sum_i I_i\sigma_i + \sum_i G(\sigma_i) \tag{3.7}$$

with $G(\sigma_i) = \int_0^{\sigma_i} g^{-1}(x)\,dx$, is a global Liapunov function for (3.5) and (3.6). The vector σ refers to the state of the neural network with components σ_i. The existence of a Liapunov function guarantees that the dynamics of the network always converges to a fixed point. However, in the discrete-time parallel dynamics L is a Liapunov function if the stability condition

$$\beta < |1/\lambda_{\min}| \tag{3.8}$$

is satisfied. Here β is the steepest slope of the function $g(x)$ and $\lambda_{\min}$ is the most negative eigenvalue of the matrix $\mathbf{T}$. If all eigenvalues are positive then L is a Liapunov function for any β. This can be understood if we calculate $\Delta L \equiv L(\sigma(t+1)) - L(\sigma(t))$:

$$\begin{aligned}\Delta L = -\frac{1}{2}\sum_{i,j} T_{ij}\left[\Delta\sigma_i\Delta\sigma_j + 2\sigma_j(t)\,\Delta\sigma_i\right] - \sum_i I_i\Delta\sigma_i \\ + \sum_i \left[G(\sigma_i(t+1)) - G(\sigma_i(t))\right]\end{aligned} \tag{3.9}$$

where $\Delta\sigma_i \equiv \sigma_i(t+1) - \sigma_i(t)$. By Taylor's theorem we obtain

$$\begin{aligned}G(\sigma_i(t+1)) - G(\sigma_i(t)) &= G'(\sigma_i(t+1))\,\Delta\sigma_i - \frac{1}{2}G''(\xi)\,(\Delta\sigma_i)^2 \\ &\le G'(\sigma_i(t+1))\,\Delta\sigma_i - \frac{1}{2}(\Delta\sigma_i)^2 \min_{x\in[0,1]} G''(x)\end{aligned} \tag{3.10}$$

where $\xi \in [\sigma_i(t), \sigma_i(t+1)]$. Inserting equation (3.10) in equation (3.9) and using the definition of $G(x)$ we obtain

$$\Delta L \le -\frac{1}{2}\sum_{i,j}[T_{ij} + \delta_{ij}/\beta]\Delta\sigma_i\Delta\sigma_j \tag{3.11}$$

where δ_{ij} is the Kronecker delta-function. Here $1/\beta = \min_x G''(x)$ is the minimum curvature of the function $G(x)$ in the closed interval $[0,1]$. The right-hand side of the inequality (3.11) is negative if the matrix $T_{ij} + \delta_{ij}/\beta$ is positive definite. A sufficient condition for matrix $\mathbf{T} + \beta^{-1}\mathbf{I}$ to be positive definite is given by equation (3.8).

In the case of the continuous-time dynamics, for each finite value of β the dynamics (3.6) converges to a fixed point which is a (local or global) minimum of the Liapunov function. By differentiation of (3.7) and using (3.6) we obtain

$$\begin{aligned}\frac{dL(\sigma(t))}{dt} &= \sum_i^N \frac{\partial L(\sigma(t))}{\partial \sigma_i}\frac{d\sigma_i(t)}{dt} \\ &= -\sum_i^N \left(g^{-1}(\sigma_i) - u_i\right)(\sigma_i - g(u_i)) \leq 0 \end{aligned} \tag{3.12}$$

since $g^{-1}(\sigma) - u$ and $\sigma - g(u)$ have the same sign. Calculating the time derivative of (3.3) we can also obtain a representation of (3.6) in terms of the input variables u_i:

$$\frac{du_i(t)}{dt} = \sum_j^N T_{ij}\, g(u_j(t)) + I_i - u_i(t) \tag{3.13}$$

The set of equations (3.13) was first proposed by Hopfield [31]. There is a subtle difference though, between (3.13) and (3.6). Hopfield assumed that the activity of neurons changes at the same time as when the neurons receive input:

$$\sigma_i(t) = g(u_i(t)) \tag{3.14}$$

In the case of dynamics (3.3) and (3.6) this assumption is only valid when the system has reached a stationary state. The dynamical system defined by (3.13) together with relation (3.14) has the same Liapunov function as (3.3) and (3.6) [31]. By direct computation,

$$\frac{dL(\sigma(t))}{dt} = \sum_i^N \frac{\partial L(\sigma(t))}{\partial \sigma_i}\frac{d\sigma_i(t)}{dt} = -\sum_i^N \left(\frac{du_i(t)}{dt}\right)^2 g'(u_i) \leq 0$$

Here we have used the symmetry of T_{ij} and the monotonic increase of the function g. The set of equations (3.13) and the Liapunov function (3.7) is a particular case of the more general dynamical system, the stability of which was studied by Cohen and Grossberg [10].

3.2.4 Feasible paths

To construct a feasible path we can use both discrete- and continuous-time dynamics given by (3.5), (3.6) or (3.13). In our simulations we have used discrete-time dynamics (3.5) for reasons of simplicity. At the initial time, $t = 0$, the

activity of all neurons is set to "zero". Under influence of an external input, which clamps the neuron at the target node to the value "one" and which clamps all neurons in the occupied nodes to "zero", the state of the system begins to change according to the dynamics of the network described by equation (3.5). The evolution of the network can be seen in the phase space S as a motion of a point along the curve on which the Liapunov function decreases. This down-hill motion ends when the network reaches an equilibrium state which is a local or global minimum of L. The network cannot display an oscillating behavior. This is guaranteed by condition (3.8). The final equilibrium states are solutions of the fixed point equations

$$\sigma_i^* = g\left(\sum_j^N T_{ij}\sigma_j^* + I_i\right) \tag{3.15}$$

for $i = 1, \ldots, N$. This final equilibrium state is unique when the Liapunov function is strictly convex. If the second order partial derivatives of L exist then L is strictly convex if and only if the Hessian matrix $L_{ij} \equiv \partial^2 L/\partial\sigma_i\partial\sigma_j$ is positive definite. In our case

$$L_{ij} = -T_{ij} + \delta_{ij}\frac{1}{g'\left(g^{-1}(\sigma_i)\right)} < -T_{ij} + \delta_{ij}/\beta \tag{3.16}$$

When all eigenvalue of the matrix T_{ij} are negative then L is strictly convex function. If some of them are positive then from sufficient condition for strict convexity is given by

$$\beta < 1/\lambda_{\max} \tag{3.17}$$

In the case of parallel discrete-time dynamics we have to satisfy simultaneously the stability condition (3.8). When we choose

$$\beta < 1/\lambda, \quad \lambda = \max\{|\lambda_{\min}|, |\lambda_{\max}|\}, \tag{3.18}$$

then the Liapunov function (equation 3.7) is strictly convex on S and at the same time condition (3.8) is satisfied. The strict convexity implies that a local minimum of the function (3.7) is a global one and is unique. In this case the set of equations (3.15) has only one solution. This equilibrium state depends only on the position of the obstacles and on the position of the target.

Consider first the situation with static obstacles and with a static target. If the neuron in the initial configuration node j_{init} is active, then a path connecting initial position and target-position is created in the following way. A new configuration for the robot manipulator is given by the position of the neighboring neuron with the largest activity, which serves then as the starting position for a new

change. Every time when a new configuration is reached a new decision is made concerning the next configuration. This procedure is repeated until the neuron corresponding to the actual position of the robot becomes the target neuron. The created path $\mathcal{P}$ passes through a sequence of points $j_{\text{init}} = j_0, j_1, \ldots, j_{\text{targ}} = j_m$. The length of the path $\mathcal{P}$ passing through these sequence of points is given by

$$D(\mathcal{P}) = \sum_{i=0}^{m-1} \rho(j_i, j_{i+1}), \tag{3.19}$$

In our case $\rho(j_i, j_{i+1})$ is equal to the lattice constant. Since the path leads from one node of the lattice to the neighboring node with the largest activity and ends in the node with activity "one", the number of steps and the length of the path on the grid is minimal. All other paths which pass through a sequence of points in which the activity does not increase monotonically or does not increase in the optimal way (i.e. largest increment), consist of larger number of steps and have a length larger than $D(\mathcal{P})$ since all steps have the same length. Due to the discretization of the configuration space, the path $\mathcal{P}$ has to stay on the nodes of the grid and is only an approximation of a smooth curve in $\mathcal{C}$. The degree of accuracy is only limited by the grain size of the grid.

3.3 Computer simulations

In this section we will illustrate some of the analytical results of the algorithm outlined above. The range of possible applications includes problems classically handled by the path planning community such as: controlling a robot arm, car manoeuvring [13] , or the piano movers problem. We have chosen four examples: an autonomous point-robot trying to find the center of a labyrinth; a two-link robot manipulator moving in a cluttered environment; a point-robot avoiding a moving obstacle in order to reach the target and a point-robot following and trying to catch up with a moving target.

In our demonstration we used a two-dimensional lattice ($d = 2$) with $N = 2500$ neurons ordered in a 50x50 neural map. So, all four examples are two-dimensional path planning problems. Note, however, that the model can easily be extended to higher dimensions and may have other types of homogeneous lattices. According to equation (3.4) each neuron is connected bi-directionally to the neurons which are lying within a distance r in the neural space around it. By choosing $r = 1.5$ each neuron has eight neighbors ($z = 8$). Each simulation starts with the activity of the neuron closest to the target position clamped to

value "one". The activities of all neurons which correspond to obstacle positions are clamped to the value "zero".

At each time-step, neurons update their activations in parallel, according to equation (3.5). The changes in neuron activations stop when the network reaches an equilibrium state.

The new actual neuron is the neuron with the highest activity, chosen from the last actual neuron and its eight neighbors. If two or more of the neighboring neurons have the highest activation then one of them is chosen randomly.

Any monotonically increasing function can be used in our algorithm as the transfer function. In the simulations we chose both $g(x) = \tanh(\beta x)$ and $g(x) = \beta x$.
To chose a value of β which satisfies the condition (3.18), we have to estimate the largest and the smallest eigenvalue of the matrix $\mathbf{T}$. Since each neuron is connected to eight neighbors, the number of nonzero elements in each row or column is at most equal to eight. The nonzero elements are all equal to one. From Frobenius theorem (see [8]) it follows then that all eigenvalues $\lambda \in [-8, 8]$. In the absence of obstacles and when we impose periodicity of the neural map, all eigenvalues are positive and the largest eigenvalue is equal to eight. The matrix $\frac{1}{8}\mathbf{T}$ in this case is a double stochastic matrix. So we have to choose $\beta \in [0, 1.25\rangle$ to guarantee the stability condition and uniqueness of the equilibrium state.

For both, the linear and tanh(.) input/output behaviour of the neurons, we chose $\beta = 0.1$. The paths generated by these functions are the same. The computational time however, is far more shorter in the linear case.

An optimal path can be found even before the network settles into an equilibrium state. In our simulations the configuration of the robot manipulator starts to change at the moment when one of the neighboring neurons of the actual neuron is activated by the target neuron. This is essential when the trajectory must be planned in a changing environment. To understand that planning can start before an stable state is reached, we consider the activity in the neural map at equilibrium. The shape of the activity in the map is a hill with the top located at the target neuron and activity zero at the obstacle neurons. In this activation pattern lines of iso-activation can be defined. The structure and shape of these lines depend only on the location of the target and obstacles. Assume that target and obstacles are fixed. At time zero only the target neuron has a non-zero activity. Initially, due to the dynamics of the neurons, all neurons located on the first iso-activation line receive input from the target neuron and are activated. In the following moment they raise their activity and send information to the neurons on the second iso-activation line. This will go on, and the activity of neurons on successive iso-activation lines is raised from zero until the activation

landscape is stable. During this dynamics the shape of the iso-activation lines is constant and the amplitude of activity decreases with increasing number of the iso-activation line. Hence, the maximal gradient on every position has a constant direction from the moment when the activation front reaches that location.

3.3.1 The labyrinth

In the first simulation we test the performance of the network in the complex environment of a labyrinth. Each configuration of a point-robot moving in a 2-dimensional workspace is described by the two position coordinates x and y. In the case of the point-robot the configuration space is isomorphic to the 2-dimensional work space. Hence the topological neural map represents a discretization of the workspace. Each neuron in it corresponds to a small unit area in the workspace. Neighboring neurons correspond to neighboring areas. By correspondence, neurons in the neural map represent the target-position, the initial actual position and the obstacles. We simulated several situations. In each case, if a continuous path between target and initial position existed, the neural net found it. In the case that more than one solution was possible, the one with the shortest path-length was chosen. In figure 3.1 we show the neural map with a set of obstacle neurons which corresponds to labyrinth walls. The initial actual position is chosen outside the labyrinth at point S and the target position is chosen in the center of the labyrinth, at point T. The obstacle neurons are represented by black dots and the neurons corresponding to the actual position in successive time-steps by open circles. Note that several paths may lead to the target position, but that the path generated by the network has the shortest length.

3.3.2 Planar robot

In this example we will consider a two-joint robot arm moving in a 2-dimensional workspace in the presence of obstacles. In contrast with the point-robot all objects have real physical dimensions. The configuration of the robot arm is described by two joint angles θ_1 and θ_2. We assume no constraints on the joints. In this case, we impose periodic boundary conditions: the neurons on the border of the map are connected to the neurons on the opposite border. The 2-dimensional map is topologically equivalent to a torus and the joint angles can vary from 0 to 2π periodically. Both joints can move simultaneously or move only one at a time. The optimal path generated by the system is given by the minimum joint motion of the robot arm in the workspace.

In workspace we can choose an initial configuration of the two-link manipula-

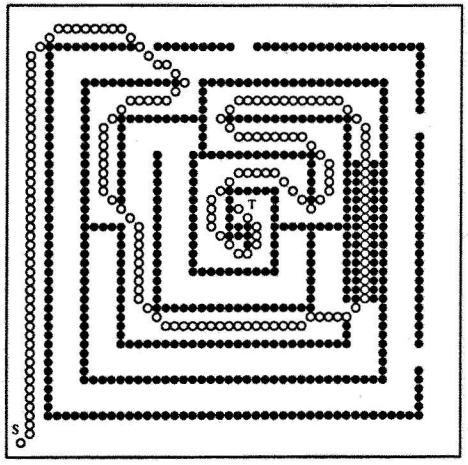


Figure 3.1: The labyrinth. Black dots represent neurons corresponding to walls of the labyrinth. The sequence of open circles gives the path generated by the network. These dots correspond also to the actual neuron at the successive time-steps. The initial position is at point S, the target-position at point T.

tor, a target-configuration and the obstacle-positions. The shape of the obstacles in the configuration space depends on the exact shape of the robot arm.

In figure 3.2(a) the workspace with the two-link manipulator is presented. The gray circles are obstacles. The manipulator has been drawn at each time-step, illustrating the movement. In figure 3.2(b) we show the corresponding neural space which is a discretization of the configuration space. The black dots represent the neurons which correspond to the forbidden configurations of the manipulator. The open circles, drawn in figure 3.2(b), show the successive actual neurons which correspond to the manipulator-configurations at every time-step.

3.3.3 Changing environment

Moving Target

Consider a situation in which the target is moving and the robot's task is to follow and to try to catch it. In this case the sensory information has to be processed continuously in order to update the network for changes in the environment. The activity of neurons will change all the time in the direction of the equilibrium configuration imposed by the instantaneous position of the obstacles and the

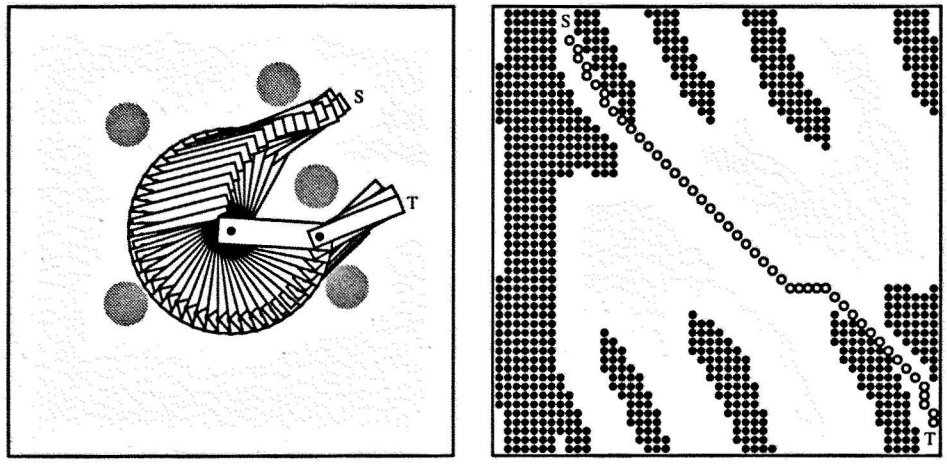


Figure 3.2: Two link robot manipulator. (a) Representation of the workspace. Gray circles represent obstacles. The robot manipulator is drawn in successive positions. (b) The configuration space. Black dots represent the neurons corresponding to the obstacles. Open circles show the actual neuron at successive time-steps.

target. The path, as in the static case, is determined by the neural activity gradient. The difference with the static environment is that the network can not settle in an equilibrium state since the neuronal representation of target and obstacles will change all the time. This will influence the direction of the gradient which will change not only from node to node but also change with time for a given node. This gradient contains the relation between neural activities and the direction of movement. A collision free path is generated only when the speed of the moving obstacles is smaller than the rate of change of neural activities.

As an example of how the network generates a path in this case, we displaced the target by one step, after each iteration. Generally, the path made by the network is always shorter than the trajectory of the target (see figure 3.3). This gives the robot the opportunity to catch up with the target, even if both have the same speed.

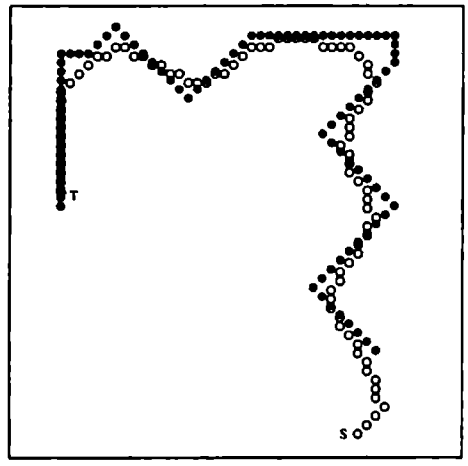


Figure 3.3: The pursuit. Black dots represent the target-position in successive time-steps. Open circles show the actual position at successive time-steps.

Moving obstacles

In addition to avoiding static obstacles, path planning must also include collision avoidance with moving objects. To illustrate this we consider the case in which the point-robot can reach the target in two ways. At the beginning of the simulation one path is blocked. During the simulation, after the robot went along a substantial part of the path (see figure 3.4(a)), we move the obstacle to obstruct this path and free the other. This causes a rapid change of neural activities and the direction of the neural activity-gradient. As a result the robot reverses and reaches the target via the other path (see figure 3.4(b)).

3.4 Conclusions

We have proposed a model for path planning and obstacle avoidance based on an analog neural network. The network is a large collection of locally and symmetrically connected elementary processors. The evolution of the network is given by parallel discrete- or continuous-time dynamics. When the network receives an external input, the neurons in the network start to change their activity towards a specific value which corresponds to a minimum of the Liapunov function. The direction of the motor response, the path, is determined by the neural activity

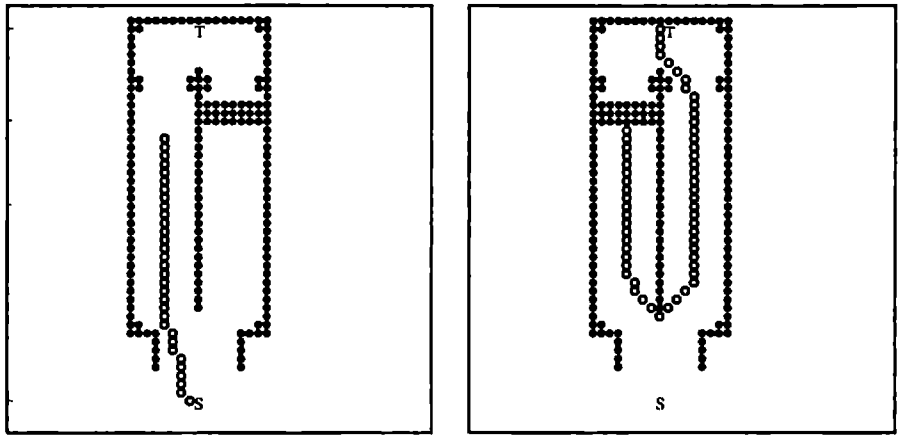


Figure 3.4: Moving Obstacle. (a) Black dots represent neurons corresponding to the labyrinth walls and the obstacle. Open circles show the actual position of the point robot at successive time-steps from the start until the moment when the obstacle is moved. (b) When the open pathway is blocked, the robot changes its direction towards the new open "gate".

gradient.

Our neural network algorithm to compute optimal feasible paths is an activity wave propagation, analogous to the Huygens principle. A distance transform method [33] is an example of this kind of algorithm used in robotics for path planning. However, our network can more easily respond to sudden changes in the environment, like moving obstacles and target, and with the possibility for hardware implementation the neural network is more powerful than the distance transform.

Our method resembles Dijkstra's algorithm of finding the lengths of the shortest paths from a given vertex (target neuron) to all the remaining vertices of the graph. However the time taken by Dijkstra's algorithm on a graph with N vertices is $O(N^2)$ and in our method is $O(N)$.

Four points are worth noting about the functional properties of the network. First, pilot computer experiments demonstrated that the system could remain functional despite of local damages in the network. The quality of the path will be somewhat degraded, but it will still be a feasible path. Secondly, we want to stress the network's insensibility to the choice of the transfer function. The system

is able to perform correctly, even if different transfer functions are used for the neural map. Thirdly, continuous neurons eliminate oscillations related to parallel dynamics of discrete neurons [48]. Fourth, the parallel architecture and dynamics of the model give it the large speed needed for real-time processing of sensory information because all neurons simultaneously and continuously change their analog states. Since the basic idea of the network can be expressed by electrical circuits, modern technologies should provide the possibilities to implement the network with a large number of processing elements in hardware. Graf et al [26] ware able to make custom chips with $N = 256$ fully connected units, using $2N^2 \approx 130,000$ resistors. Since in our network all connections are positive and local the number of resistors required will be of the same order as the number of neurons in the network.

Chapter 4

A Biologically Inspired Neural Net for Trajectory Formation and Obstacle Avoidance

Abstract

In this paper we present a biologically inspired two-layered neural network for trajectory formation and obstacle avoidance. The two topographically ordered neural maps consist of analog neurons having continuous dynamics. The first layer, the sensory map, receives sensory information and builds up an activity pattern which contains the optimal solution (i.e. shortest path without collisions) for any given set of current position, target positions and obstacle positions. Targets and obstacles are allowed to move, in which case the activity pattern in the sensory map will change accordingly. The time-evolution of the neural activity in the second layer, the motor map, results in a moving cluster of activity, which can be interpreted as a population vector. Through the feedforward connections between the two layers, input of the sensory map directs the movement of the cluster along the optimal path from the current position of the cluster to the target position. The smooth trajectory is the result of the intrinsic dynamics of the network only. No supervisor is required. The output of the motor map can be used for direct control of an autonomous system in a cluttered environment or for control of the actuators of a biological limb or robot manipulator. The system is able to reach a target even in the presence of an external perturbation.

Computer simulations of a point robot and a multi-joint manipulator illustrate the theory.

Adapted from: R. Glasius, A. Komoda and C.C.A.M. Gielen, A Biologically Inspired Neural Net for Trajectory Formation and Obstacle Avoidance, *Biological Cybernetics*, 84: 511-520, 1996.

4.1 Introduction

One of the main themes of research on robot control deals with the issue of trajectory planning and formation. The aim of this research is to develop algorithms which allow autonomous systems or manipulators to move in a cluttered environment, which may change as a function of time, to one of multiple targets along a path without collision.

The first steps to deal with this problem were based on global methods which can be considered as a search process for a path in a graph which represents the accessible paths along objects [4]. This approach is practically useful only in a static environment, since the time to construct a graph and to perform the planning task becomes excessively long for a large number of obstacles.

As an alternative the idea of potential fields was proposed in which the target acts as an attractor in the midst of obstacles represented by repelling potentials [36, 42]. Unfortunately, the potential field methods suffer from several problems, one being that of undesired local minima. Although several suggestions have been presented to circumvent these problems [53, 75, 5, 11, 3], no algorithm was presented which guarantees a solution. Glasius et al. (1994) proposed a neural network with continuous time dynamics that also suffers from local minima. However, it improved upon previous models in the quality of the path which is smooth and continuous and which simply results from the intrinsic dynamics of the network only. The basic elements of this model are similar to the neural fields model of Amari [2] (see also [40]).

In another approach a neural-network type was proposed [47, 66, 13] in which neurons with lateral interactions are positioned on the nodes of a grid. Due to lateral interactions activity from the target spreads through the network (except towards neurons representing obstacles). The optimal trajectory to the target is found by the sequence of neurons which is obtained by stepping along the gradient ascent to the peak of activity in the network, i.e. the target. Although this procedure is shown to give the optimal path, it is not an unsupervised method, since it requires an external observer to compare the activities of neighboring neurons in order to decide which neuron should be the next in the path towards the target. Recently, variations of this distance-transform or wave-propagation method have been proposed [58, 24] using topographically ordered maps.

Although supervised methods can be useful for applications, they cannot be used for biologically plausible models for trajectory planning and formation. However, unsupervised methods have been reported for topographically ordered maps and hierarchically organized topographically ordered neural networks are frequently found in the nervous system. With the local learning rule, these prop-

erties give the models by Glasius et al. (1994,1995) and Prassler (1989) a biologically plausible basis. In this paper we present a neural network which is a composition of our two previous models. Contrary to the previous models it does not suffer from local minima and does not require an external observer. Our model guarantees a trajectory from any initial position to a target (if one exists) given an arbitrary environment with obstacles, which are allowed to change position. The trajectory is the result of internal dynamics of two layers of neurons with feedforward connections and is the shortest path in terms of the metric of the representation [43]. The biological features that are contained in our model (i.e. layered topographic maps, large numbers of continuous valued neurons, analog dynamics and the use of neighborhood functions) result in a robust model which generates smooth and continuous paths.

This paper is organized in the following way. First we will describe the model and its dynamics. The theoretical results will be supported by numerical computer simulations, which demonstrate the performance of the model for non-trivial problems of trajectory planning. In the discussion we will compare the results of the model with those of previous models. We will also discuss to what extent the model can be used as a biologically plausible model which captures the results on trajectory planning in the neurophysiological literature and which can be used to propose some new hypotheses.

4.2 The Model

The model which we will present, can be used for path planning and trajectory formation in a finite (D-) dimensional state space of any system. The state space $\mathcal{C}$ can be a Cartesian work space or the configuration space of a multi joint manipulator. Let us regard a topographically ordered neural map in which the activity of each neuron i is related to a subset in $\mathcal{C}$, called the receptive field rf_i of neuron i. As in visual neurophysiology, where the receptive field refers to that part of visual space, where visual stimuli will affect the response of a neuron, we will use the term "receptive field' as that part in stimulus space, which affects the responses of the (artificial) neurons in this study. Related to the topographical ordering, neighboring neurons have neighboring receptive fields in $\mathcal{C}$, which in general will have some overlap. For the performance of the type of network, presented in this paper, the precise shape of the receptive field is not important [2] and in our simulations we will use different shapes of receptive field. Each receptive field rf_i can be represented by a D dimensional vector $\boldsymbol{\theta}_i \in \mathcal{C}$ which points to the center of the receptive field. Hence, to each neuron i corresponds a

receptive field rf_i and a vector $\boldsymbol{\theta}_i$. If the receptive fields cover the complete state space

$$\mathcal{C} = \bigcup_{i=1}^{N} \mathrm{rf}_i \ , \tag{4.1}$$

then the grid, consisting of all $\boldsymbol{\theta}_i$, is a discrete representation of the state space $\mathcal{C}$.

Topographically ordered neural maps can be obtained in several ways (for example by using Kohonen's learning rule for topographically ordered maps [39, 59] or by a vector-quantization learning rule [59, 17, 49]). Such topographically ordered maps are used in both layers of our neural network model. The first will be denoted as the "sensory map", because it is assumed to represent sensory information about the environment of the system. The other map will be referred to as the "motor map", because its activity codes action commands.

To distinguish the variables and parameters between the two maps, the variables and parameters in the sensory map will have indices i and j and those in the motor map indices k and l.

The sensory map projects to the motor map with feedforward connections which will be defined later (see Eq. 4.8). For a two-dimensional system the architecture of the neural network containing the topographical maps and the connections are illustrated schematically in figure 4.1.

The activities of the neurons in both maps are characterized by real-valued variables

$$\sigma_i, \sigma_k \in [0, 1]$$

where σ_i denotes the activity of neuron i in the sensory map and σ_k that of neuron k in the motor map.

The input to a neuron, the local field, is defined as the weighted sum of the activities of all neurons in the network which project to that neuron minus a threshold. The local field for neurons is denoted in the sensory map by u_i and for neurons in the motor map by u_k. The activity of a neuron is given by the value of a sigmoid function on the local field. Our choice for the sigmoid function is

$$g(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ \beta x & \text{if } x \in [0, 1] \\ 1 & \text{if } x \geq 1 \end{cases} \tag{4.2}$$

with $\beta \in [0, 1]$. However, it could be any bounded function between 0 and 1 which is monotonically increasing. Simulations demonstrated that the results of this study did not depend on the type of sigmoidal function.

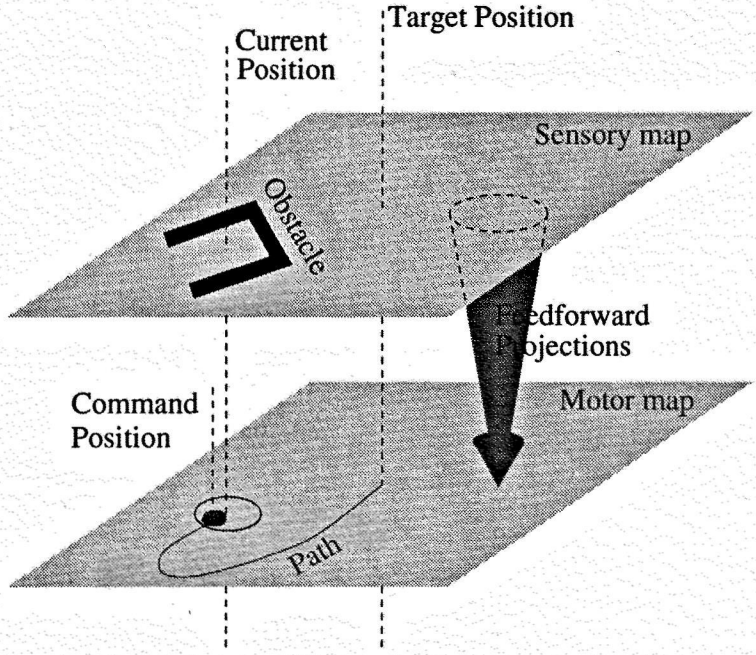


Figure 4.1: A schematic representation of the two-dimensional sensory map and motor map as a two-layer feedforward neural network.

Consequently, the neuronal activities are

$$\begin{aligned} \sigma_i(t) &= g(u_i(t)) \qquad \text{for neurons in the sensory map} \\ \sigma_k(t) &= g(u_k(t)) \qquad \text{for neurons in the motor map} \end{aligned}$$

4.2.1 The Sensory map

We assume lateral excitatory short-range connections B_{ij} from neuron j to neuron i in the sensory map:

$$B_{ij} = f(|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j|) \tag{4.3}$$

where $f(x)$ is a monotonically decreasing neighborhood function and $|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j|$ is the Euclidean distance in state space between $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$. For example a commonly

used [46, 44, 45] function is

$$f(x) = \begin{cases} e^{-\gamma x^2} & \text{if } |x| \in [0, r] \\ 0 & \text{if } |x| > r \end{cases} \tag{4.4}$$

with γ and r real positive numbers.

External input is assumed to provide information about target and obstacle positions. The sensory neuron the receptive field of which contains the target is called the target neuron. The state of the target neuron t is supposed to be equal to the value $\sigma_t \approx 1$, i.e. maximally activated. This can be achieved by a large input from external sensory neurons through excitatory connections. For simplicity we will assume that the size of a target position is smaller than the size of a receptive field such that there is only one target neuron at each moment in time. The set of positions or the states in C, which cannot be reached because of obstacles, will be represented by $\{S\}$. $\{S\}$ is defined both by the shape of the obstacles as well as by the shape of the robot. The set of neurons $i \in \{S\}$ is supposed to be de-activated by a large inhibitory input from external sensory inputs, such that these neurons have a low activity, i.e. $\sigma_{i \in \{S\}} = 0$. Because target and obstacles may move as a function of time, the neurons, which represent target and obstacles may change continuously.

In conclusion, the external sensory input to neuron i in the sensory map is given by

$$I_i = \begin{cases} v & \text{if target} \in \text{rf}_i \\ -v & \text{if obstacle} \in \text{rf}_i \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

where v is a positive number, large enough to dominate over all other inputs. The threshold of the neurons in the sensory map is set to zero.

The total input to a neuron in the sensory map is given by

$$u_i(t) = \sum_j^N B_{ij}\, \sigma_j(t) + I_i$$

The dynamics for the neurons in the sensory map can be formulated by the change of the neural input u_i per unit of time

$$\frac{du_i(t)}{dt} = \sum_j^N B_{ij}\, \sigma_j(t) + I_i - u_i(t) \tag{4.6}$$

The function

$$L(\sigma_i) = -\frac{1}{2}\sum_{i,j} B_{ij}\,\sigma_i\,\sigma_j - \sum_i I_i\sigma_i + \sum_i G(\sigma_i)$$

with $G(\sigma_i) = \int_0^{\sigma_i} g^{-1}(x)\,dx$, is a global Liapunov function [31, 27, 24] for the sensory map. The existence of a Liapunov function guarantees that the dynamics of the sensory layer always converges to a fixed point.

Due to the excitatory local interactions, the target neuron, kept at a high activity, feeds its nearest neighbors through the short range connections, such that the activity spreads through the network. As shown in a previous paper ([24]) the activation landscape which arises in the sensory neural map, is shaped like a hill with its top at the target neuron and with valleys at the locations of the obstacle neurons. As we proved in our paper [24] this activation landscape is unique and the condition for equilibrium $\frac{du_i(t)}{dt} = 0$ guarantees that no plateaus can exist. Therefore, for each given initial position, target position and the set of obstacles, the sensory map contains the optimal solution [24] constructed by successive gradient ascents to the top.

This neural activity field is passed to the motor map neurons, and, in addition, neurons in the motor map receive an external input which codes the current state of the system. The motor map has to construct a motor command which will alter the current position in a direction along the path towards the target.

In the case of moving objects, the set of obstacle neurons and the set of target neurons may change continuously in time. The activation landscape in the sensory map changes accordingly and if the obstacles move slowly relative to the time constant of the system dynamics, then the sensory map will always contain adequate information to the path planning problem.

4.2.2 The Motor map

The motor map receives inputs through feedforward connections from the sensory map. In addition, the motor map receives input which provides information about the current state of the system. This input could originate from sensors in the actuators of the limb or manipulator. With both input signals the motor map is able to produce motor commands which will continuously change the current position of the actuator such that it moves from the initial position to the target position.

The motor map has both short range and long range lateral interactions. The short range excitatory interactions are the same as those in the sensory system:

$$B_{kl} = f(|\theta_k - \theta_l|)$$

The lateral long-range interactions are inhibitory with strength

$$J_{ij} = \begin{cases} J_0/N & \text{if } i \neq j \text{ , with } J_0 \in \mathbb{R}^+ \\ 0 & \text{otherwise} \end{cases}$$

with N representing the number of neurons in the motor map. A motor neuron k has a constant threshold of

$$J_0\mu - \frac{\sum_l B_{kl}}{2} - \alpha \tag{4.7}$$

where $\mu \in R$ and $\alpha \in R^+$ are constants. ¿From formula (4.7) one might conclude that each neuron k may have a different threshold. However, when the neurons are distributed homogeneously in state space, all neurons have the same number of neighbors. This results in all neurons in the motor map having the same threshold. For neurons at the boundary of the map the threshold, which depends on the number of neighboring neurons (summation over index l in Eq. (7)) has been adjusted in order to correct for the smaller number of neighboring neurons.

Each state $\boldsymbol{\theta} \in \mathcal{C}$ of the actuator is represented in the sensory map and in the motor map. The connections from the sensory neuron i to the motor neuron k are given by

$$\hat{B}_{ki} = \frac{\alpha}{\sum_j B_{kj}} \hat{f}\left(|\boldsymbol{\theta}_k - \boldsymbol{\theta}_i|\right) \tag{4.8}$$

where α is the same constant as in (4.7). $\hat{f}(x)$ has the same shape as $f(x)$ but differs in the steepness $\hat{\gamma} \neq \gamma$ and the radius $\hat{r} \neq r$.

These connections provide part of the input signals to the motor map. The other part has an external origin. It provides an input with magnitude

$$K_k\left(\boldsymbol{\theta}_A\right) \equiv -v + v\, \tilde{f}\left(|\boldsymbol{\theta}_A - \boldsymbol{\theta}_k|\right) \tag{4.9}$$

where $\boldsymbol{\theta}_A$ represents the "current manipulator position" in $\mathcal{C}$ and where $\tilde{f}\,(x)$ is a similar function as $f\,(x)$ with $\tilde{\gamma} < \gamma$ and with $\tilde{r} > r$. The large inhibitory input $-v$ suppresses all other inputs to neurons in the motor map with a receptive field at a distance larger than $\tilde{r}$ from the current manipulator position. Therefore, the activity of these neurons is close to the minimal output value of the sigmoid function g.

With this particular combination of inputs to the motor map the activity in the motor map is grouped in a cluster with an almost constant size. The cluster of activities has to move in the map in a direction supplied by the inputs from the sensory map. In that case the neural activity in the motor map can be interpreted

in terms of position of the manipulator. By defining the following macroscopic quantities we raise our focus from the neuronal to the ensemble representation. We define the "mean total activity" m

$$m \equiv \frac{1}{N} \sum_k \sigma_k \tag{4.10}$$

Then we define $\boldsymbol{\theta}_B$ as the "command manipulator position"

$$\boldsymbol{\theta}_B(t) \equiv \frac{\sum_k \boldsymbol{\theta}_k \, \sigma_k(t) \, \tau_k(t)}{\sum_k \sigma_k(t) \, \tau_k(t)} \tag{4.11}$$

with $\tau_k(t) \equiv [1 + \operatorname{sgn}(\sigma_k(t) - \xi)]$. This is the weighted mean of all centers of receptive fields of motor neurons that have an activity larger than the threshold ξ. A neuron belongs to this set if τ_k is nonzero.

The vector $\boldsymbol{\theta}_B(t)$ is the output of the system which is sent to the actuators as a command to realize this position in $\mathcal{C}$. However, it takes some time before the actuators receive the command signals and before it moves the current position to that state. Therefore, the command manipulator position points to what will be the current position after a particular time delay. The command position always leads the current position until it finally reaches the target position.

Because the target position $\boldsymbol{\theta}_T(t)$ may move in time, more information can be provided using relative coordinates with respective to the target position instead of absolute coordinates. Therefore we define the "**relative** command manipulator position" $\mathbf{X}$

$$\mathbf{X} \equiv \boldsymbol{\theta}_B - \boldsymbol{\theta}_T$$

Similarly, this vector can be defined as a weighted mean:

$$\mathbf{X}(t) \equiv \frac{\sum_k \mathbf{x}_k(t) \, \sigma_k(t) \, \tau_k(t)}{\sum_k \sigma_k(t) \, \tau_k(t)} \tag{4.12}$$

with $\mathbf{x}_k(t) = \boldsymbol{\theta}_k - \boldsymbol{\theta}_T(t)$ as the position in $\mathcal{C}$ of motor neuron k relative to the target position. The relative command manipulator position $\mathbf{X}$ is the weighted mean of all relative vectors of motor neurons that have an activity larger than a certain activity threshold ξ. In this perspective we can regard this variable $\mathbf{X}$ as a population vector [9, 18, 34].

The total input to a motor map neuron is given by

$$u_k(t) = \sum_l B_{kl}\sigma_l - J_0(m - \mu) - \alpha\left(1 - \frac{\sum_i B_{ki}\sigma_i}{\sum_i B_{ki}}\right) + K_k(\boldsymbol{\theta}_A) - \frac{\sum_l B_{kl}}{2}$$

As explained before [23] each term in this expression has a different effect on the behavior of the network. The term $J_0(m-\mu)$ tends to set the number of active units such that $m = \mu$. The local neighbor interactions $\sum_l B_{kl}\sigma_l$ make it more advantageous for neighboring units to be active. Therefore, this term tends to impose clustering of the active units and causes that the cluster avoids the obstacle. Input from the sensory map causes the cluster to move to the maximum of that input which is the target position ($\sigma_i \approx 1$ in the sensory map). The external input $K_k(\theta_A)$ makes sure that the cluster, i.e. the command position, is near the current position (4.9).

The result of the combined interaction of the terms, is a cluster of activities in the neural motor map, moving continuously and smoothly in the direction of the gradient of the local dragging field.

If we make the following substitutions

$$\begin{aligned} T_{kl} &\equiv B_{kl} - \frac{J_0}{N} \\ I_k &\equiv J_0\mu - \alpha\left(1 - \frac{\sum_i B_{ki}\sigma_i}{\sum_i B_{ki}}\right) + K_k(\theta_A) - \frac{\sum_l B_{kl}}{2} \end{aligned} \tag{4.13}$$

then the equation for the input of the motor neurons is similar to the equation for the input of the sensory neurons.

$$u_k(t) = \sum_l^N T_{kl}\,\sigma_l(t) + I_k$$

Hence, the equations describing the dynamics for the neurons in the motor map are similar to that in the sensory map

$$\frac{du_k(t)}{dt} = \sum_j^N T_{kl}\,\sigma_l(t) + I_k - u_k(t) \tag{4.14}$$

resulting in a similar expression for the Liapunov function,

$$L(\sigma) = -\frac{1}{2}\sum_{k,l} T_{kl}\,\sigma_k\,\sigma_l - \sum_k I_k\sigma_k + \sum_k G(\sigma_k)$$

with $G(\sigma_k) = \int_0^{\sigma_k} g^{-1}(x)\,dx$

The system has the ability to react to external forces acting on the manipulator. If an external force suddenly changes the state of the system, the external input $K_k(\theta_A)$ changes accordingly and causes an inhibition for all neurons except for those in the neighborhood , characterized by $\tilde{r}$ of $\tilde{f}$ in (4.9), around the new

current position. If the command position $\mathbf{X}_B$ is outside this region then the cluster disappears due to the inhibition and reappears at the location with the highest local field [24]. Therefore, a change of the current position is followed by a change in the command position which can be interpreted as a reaction to the perturbation in order to adapt the path to the new situation.

If the external force is continuously present, the movement of the current position is not only due to the actuators but also due to the external force. The cluster, however, is always trapped in the vicinity of the current position, i.e. the area defined by $\tilde{r}$ of $\tilde{f}$. Inside this area the cluster is always located near the neurons with the highest local fields, i.e. in the direction along the optimal path to the target position. Hence if the force produced by the actuators is strong enough to dominate over the external force then the target will be reached.

The length $\tilde{r}$ depends on the time delays in the motor system and on the velocity of the manipulator. It takes some time to send the motor command from the motor map to the actuators, to move the actuator and to send the new current position back to the motor map. Consequently the current position is always a period of time Δt lagging behind the command position. Depending on the velocity of the cluster in the motor map, as a result of the system dynamics, the length $\tilde{r}$ in state space can be calculated such that the command position is never further away than the cluster can move in a time Δt.

4.3 Computer simulations

4.3.1 A point robot in a two dimensional work space

In the first simulation we regard a point robot moving in a square subset of a two-dimensional Cartesian work space. The initial position of the point robot is in the cavity of a non-convex obstacle (see figure 4.2). The target position is placed at the other side of the obstacle.

The sensory map and the motor map are both two-dimensional neuronal maps, consisting of a square lattice of 50x50 neurons. In both maps the receptive fields cover the complete work space according to equation (4.1) and both have the same topography as the work space.

As found in biology the lateral connection strength decreases with the distance between two neurons and the neurons do not project to themselves. The short range lateral connections can be characterized by the neighborhood function

$$f(x) = \begin{cases} \mathrm{e}^{-(x-1)^2} & \text{if } x \in \langle 0, 3] \\ 0 & \text{if } x = 0 \text{ and } x > 3 \end{cases}$$

Note that x is discrete, corresponding to the distances between the nodes on the lattice. Larger values than $r = 3$ could have been chosen without loss of functionality, since the exponential function becomes small for $r = 3$. In order to limit the computational time we chose the small range of $r = 3$.

Having defined the neighborhood in the maps, we assume that sensory neurons project to motor map neurons with receptive fields in the same neighborhood. Again values larger than $\hat{r} = 3$ can be chosen but will only increase the computational time.

$$\hat{f}(x) = \begin{cases} 1 & \text{if } x = 0 \\ e^{-(x-1)^2} & \text{if } x \in (0, 3] \\ 0 & \text{if } x > 3 \end{cases}$$

For the projections of sensory inputs from external origin to the motor map (equation (4.9)) we chose the neighborhood function

$$\tilde{f}(x) = \begin{cases} e^{-0.01x^2} & \text{if } x \in [0, 6] \\ 0 & \text{if } x > 6 \end{cases}$$

The functionality of the model is insensitive to the choice of $\tilde{r}$. The value $\tilde{r} = 6$ adheres to the constraint that $\tilde{r} > r$ and corresponds to a particular Δt. The external inputs to the sensory map are described in equation (4.5).

Larger values for r, $\hat{r}$ and $\tilde{r}$ imply that each neuron has a larger number of connections. In addition to the larger amount of computational time in the simulations, these larger values make the system less sensitive to broken connections or noisy neurons.

Because the input to target and obstacle neurons must be large such that target neurons in the sensory map will have an activity $\sigma_i = 1$ and obstacle neurons in the motor map will have an activity $\sigma_k = 0$, we chose $v = 1000$ (see eq. (4.5) (4.9)). If we choose β in the transfer function (4.2) high, then a non-target sensory neuron with low input can have a large output. Moreover, it can have an output $\sigma_k = 1$ which, in the following moment, will be passed to its neighbors. The result is that all neurons will end up in maximal activated state. If we choose β small then the activity of a neuron will be low, even when its input is high (e.g. when the neuron is a neighbor of the target neuron). In that case the hill in the activity landscape will decrease rapidly to activity values close to zero. ¿From trial and error we found that $\beta = 0.3$ is, in this case, just small enough to prevent the system to blow up, and large enough to have a broad hill on the activity landscape. In the simulations we used a discrete-time version of

the dynamics (4.6). At each time step dt all neurons change their state according to

$$\begin{cases} u_i(t+dt) = u_i(t) + dt\left(\sum_j^N B_{ij}\,\sigma_j(t) + I_i - u_i(t)\right) & \text{sensory neuron} \\ u_k(t+dt) = u_k(t) + dt\left(\sum_j^N T_{kl}\,\sigma_l(t) + I_k - u_k(t)\right) & \text{motor neuron} \end{cases}$$

In the sensory map a peak of activity around the target neuron grows in width. The target neuron, activated by external input to its maximum, activates its neighbors through the short range connections. The neighbors of the target neuron activate their neighbors and so on. In the vicinity of an obstacle neuron, the activity of which is kept at the minimal value, the spread of activities stops in the direction of the obstacle neuron. The resulting neural activity values in the sensory map after some time are shown in figure 4.3. If target nor obstacle move, then the resulting hill is stable.

The neural activity in the motor map at four points in time is shown in figure 4.4.
At all times there is a cluster localized somewhere in the map. Initially the cluster is located at the start position, i.e. in the cavity of a non-convex obstacle. Then, as a result of the system dynamics only, the cluster moves away from the target, it rounds the obstacle and eventually it moves to the target and reaches it. The path is smooth and continuous which has been illustrated in figure 4.2.

4.3.2 Simple connections for real time robot trajectory formation

Consider a D-link robot manipulator. In this case the state space is a D-dimensional configuration space and can be described by the D joint angles of the links. The structure of both topographical maps is a D-dimensional cubic lattice of neurons. The neurons have a D-dimensional square shaped receptive field that does not overlap with the receptive field of other neurons. The receptive fields are contiguous and the map is periodic such that they cover the complete configuration space. Therefore, the centers of the receptive fields are also ordered according to a D-dimensional lattice.

The neighborhood functions in the lateral short range connections of the two topological maps are

$$f(x) = \begin{cases} 1 & \text{if } x \in \left\langle 0, \sqrt{2} \right] \\ 0 & \text{otherwise} \end{cases}$$

resulting into lateral short range connections that are symmetric with value 1. Every neuron i in the sensory map projects to only one neuron k in the motor

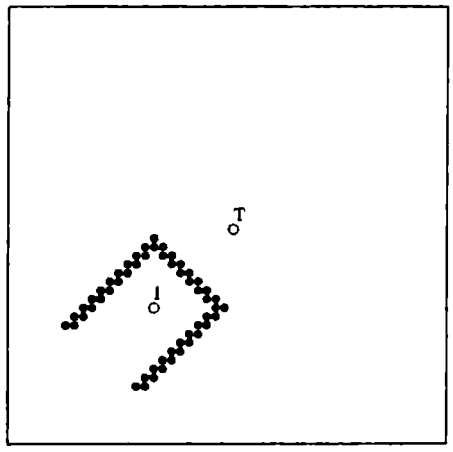


Figure 4.2: Representation of work space. Black dots are part of a non-convex obstacle. The open circle marked with an I is the initial position. The open circle marked with a T is the target position. The line illustrates the smooth and continuous path generated by the model.

map with the same receptive field

$$\hat{f}(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases}$$

The neighborhood function, characterizing the projections from external origin, has been chosen:

$$\tilde{f}(x) = \begin{cases} 1 & \text{if } x \in [0, 6] \\ 0 & \text{otherwise} \end{cases}$$

The feedforward projections from the sensory map to the motor map consist of connections with strength α between the corresponding neurons. Let $\sigma_{i(k)}$ be the state of sensory neuron i which projects to neuron k in the motor map, then the input from the first layer is

$$-\alpha\left(1 - \sigma_{i(k)}\right) \tag{4.15}$$

Conclusively, the input to a sensory neuron is

$$u_i(t) = \sum_{j}^{N} B_{ij}\, \sigma_j(t) + I_i$$

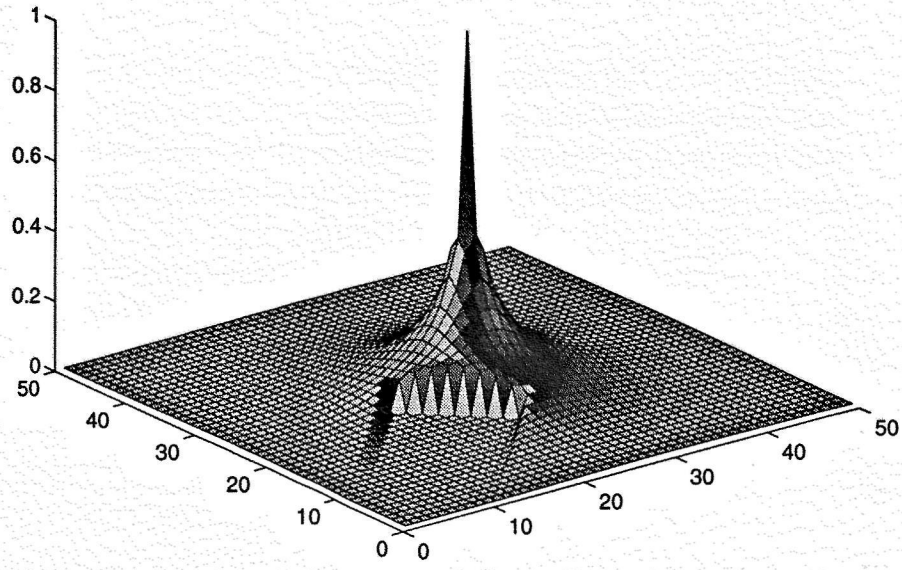


Figure 4.3: Stable neural activity in the sensory map as function of the position of the neurons in the 50x50 neural grid (See section 3.1). The map is isomorphic to the work space in figure 4.2.

and to a motor neuron

$$u_k(t) = \sum_l B_{kl}\sigma_l - J_0(m-\mu) - \alpha\left(1-\sigma_{i(k)}\right) + K_k(\theta_A) - \frac{\sum_l B_{kl}}{2}$$

The long range connections in the motor map provide a global field. Each neuron temporarily receives the same amount of input through the lateral connections. The number of connections per neuron in the motor map can be decreased dramatically by using an extra external integrating neuron. The N long range connections to the other neurons can be replaced by one connection of the original strength J_0/N to the external neuron and one connection from this neuron back with weight one.

In the simulation we chose $\alpha = 1$ and $D = 2$. The results, however, do not critically depend on the value of these parameters. With a square lattice the last term is now $\frac{\sum_l B_{kl}}{2} = 4$. The other parameters as well as the initial configuration, the target configuration and the obstacle configuration are chosen the same as in the former simulation (see figure 4.2).

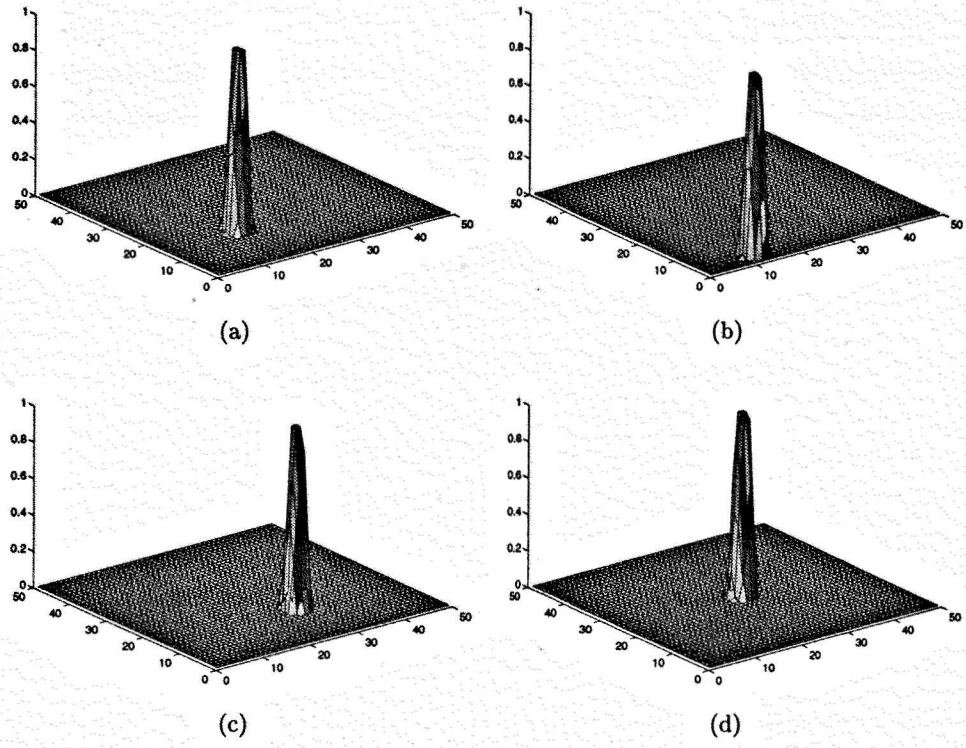


Figure 4.4: Activation landscape in the motor map at four time instances (a, b, c, and , respectively) during the movement of the cluster to the target position (See section 3.1). The neuronal map is isomorphic to the work space in figure 2.

The stable state of neural activities in the sensory map is shown in figure 4.5(a). The cluster in the motor map located at the target configuration is shown in figure 4.5(b). The differences with the results of the first simulation, due to the simplified connections, are the width of the activity hill in the sensory map, which is smaller than that in figure 4.3, and the number of motormap neurons inside the cluster which is larger than in figure 4.4(d).

The speed of the cluster movement depends on the slope of the activation hill. Because the slope of the activation hill in the outer region is smaller than in the previous simulation, the speed in system-time units is smaller in this simulation compared to the previous simulation. However because of the simple connections

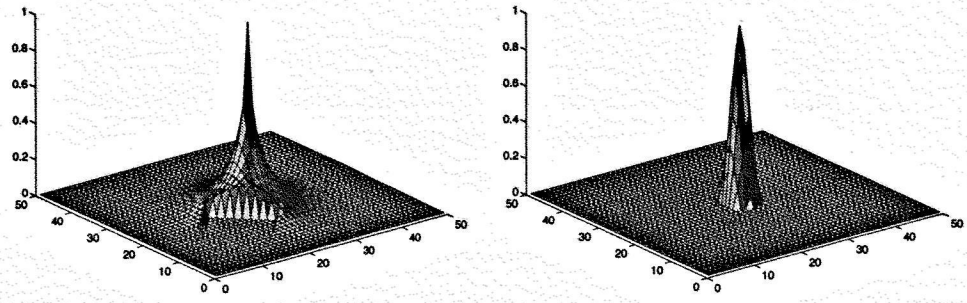


Figure 4.5: Stable neural activity as a function of the position of the neurons in the 50x50 neural grid (see section 3.2). Both neural maps are isomorphic to the work space in figure 4.2. a) The sensory map. The width of the activation hill is smaller than the width of the activation hill in figure 4.3. b) The motor map. The cluster is at target position. The cluster has the same mass of positive activity on an area larger than that in figure 4.4(d).

the time to simulate this system on a sequential computer (DEC 3100) is about an hour while that of the first simulation is a few days.

A more localized cluster has the same total mass of positive activity on a smaller area, which means that the cluster is more dense. The density of the cluster is important when it is in the vicinity of an obstacle. A dense cluster has higher neural activities which results in a better preservation of the spherical shape of the cluster.

Nevertheless, the resulting paths in the two simulations are qualitatively equal and we want to stress that the model is quite insensitive to the choice of the neighborhood functions.

4.3.3 Choosing between multiple targets

In the following simulations we regard a two link manipulator working in a two-dimensional work space. To reach any point in work space with the manipulator end-effector two configurations are possible. Hence the system has to chose between two target manipulator configurations having the same end-effector work space coordinates. The simulations will be done under the two conditions: with

and without obstacles in work space.

The configuration space of the manipulator can be described by the two joint angles of the manipulator. The neural maps, again, consist of a neural grid of 50 by 50 neurons. Each neuron has a hyper-cube shaped receptive field. Because the joint angles are periodic the neuronal maps have to be cyclic. Hence the shape of both maps is that of a torus.

The other parameters are chosen the same as in the second simulation described in section 3.2 of the point robot with simple connections.

The resulting neural activity on the sensory map is shown in figure 4.6(a). Depending on the initial position of the manipulator in configuration space, steepest ascent will lead to one of the tops. Both target configurations have a basin of attraction consisting of all initial configurations that will result in a stable end-configuration at that top.

In the first simulation we chose two initial configurations, one leading to one top and the other to the other top. In figure 4.7(a) and 4.7(b) the manipulator is shown at four different time instances.
Note that the end-effector target configuration in both pictures is the same. In figure 4.7(c) both paths are shown in configuration space.

In the second simulation, all manipulator configurations which are blocked by the obstacle are called obstacle configurations. The neurons with an obstacle configuration in its receptive field are referred to as the obstacle neurons. The obstacle neurons block the activity flow from both target neurons. The resulting activity landscape in the sensory map is shown in figure 4.6(b). Note that the basin of attraction belonging to the target configuration far from the obstacle is enlarged, while that of the near target configuration became smaller. We chose an initial configuration close to the target configuration near to the obstacles. The resulting manipulator movement is illustrated in figure 4.7(d), the path in configuration space is shown in figure 4.7(e). Note that in the former simulation this initial configuration would lead to the other target configuration.

4.4 Discussion

In this paper we proposed a model for trajectory formation and obstacle avoidance based on a two-layered neural network with continuously-valued neurons. The evolution of the network is given by parallel continuous-time dynamics. The advantage of the two-layered network is that it allows trajectory formation in a cluttered environment without an explicit algorithm how to reach the object in work or joint space. This was not possible in a single-layer network.

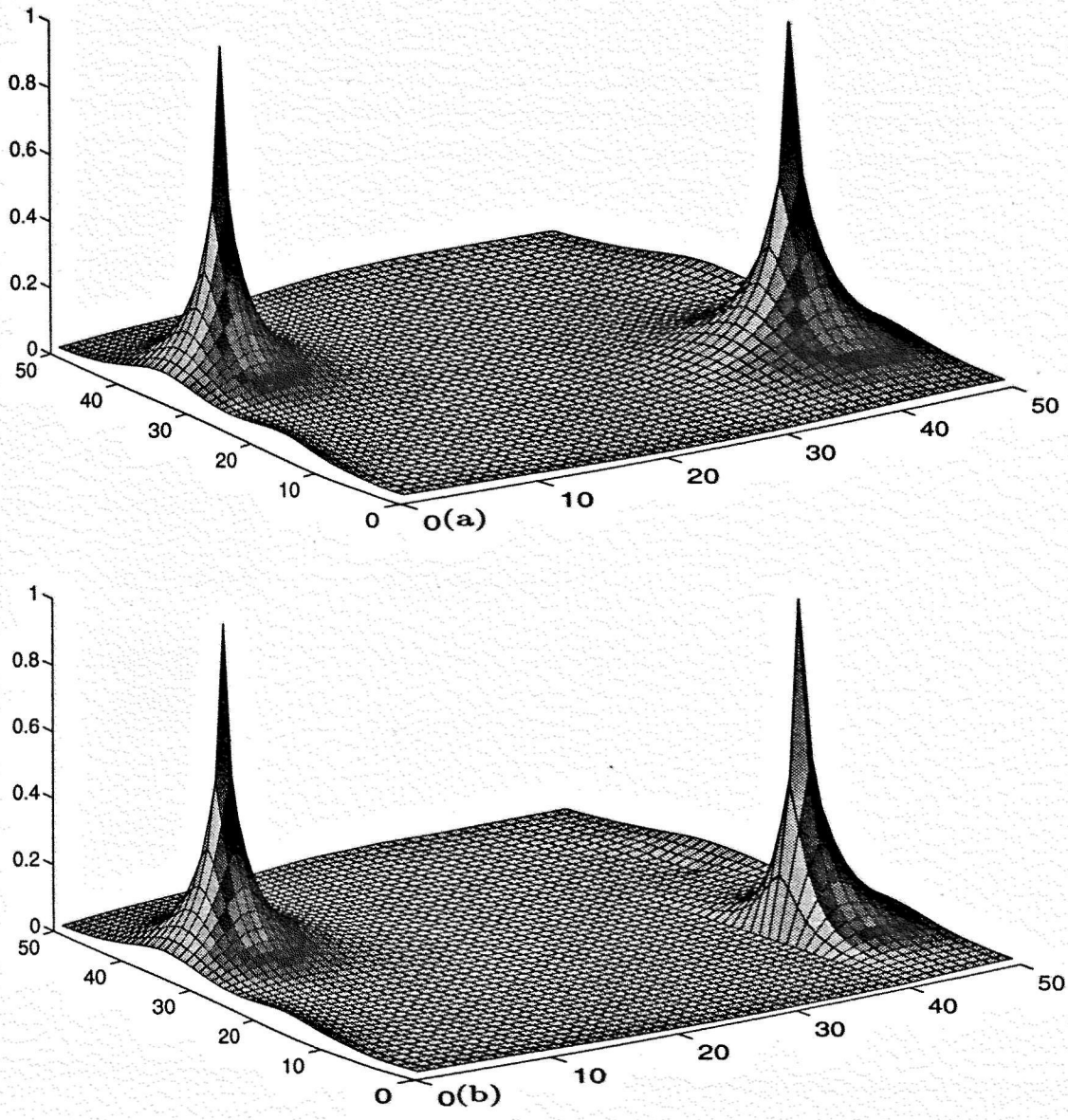


Figure 4.6: Stable neural activity in the sensory map as a function of the position of the neurons in the 50x50 neural grid. a) Two target positions in configuration space correspond to the same manipulator endpoint coordinate in work space. b) An obstacle in work space changes the flow of activities in the sensory map and the resulting landscape.

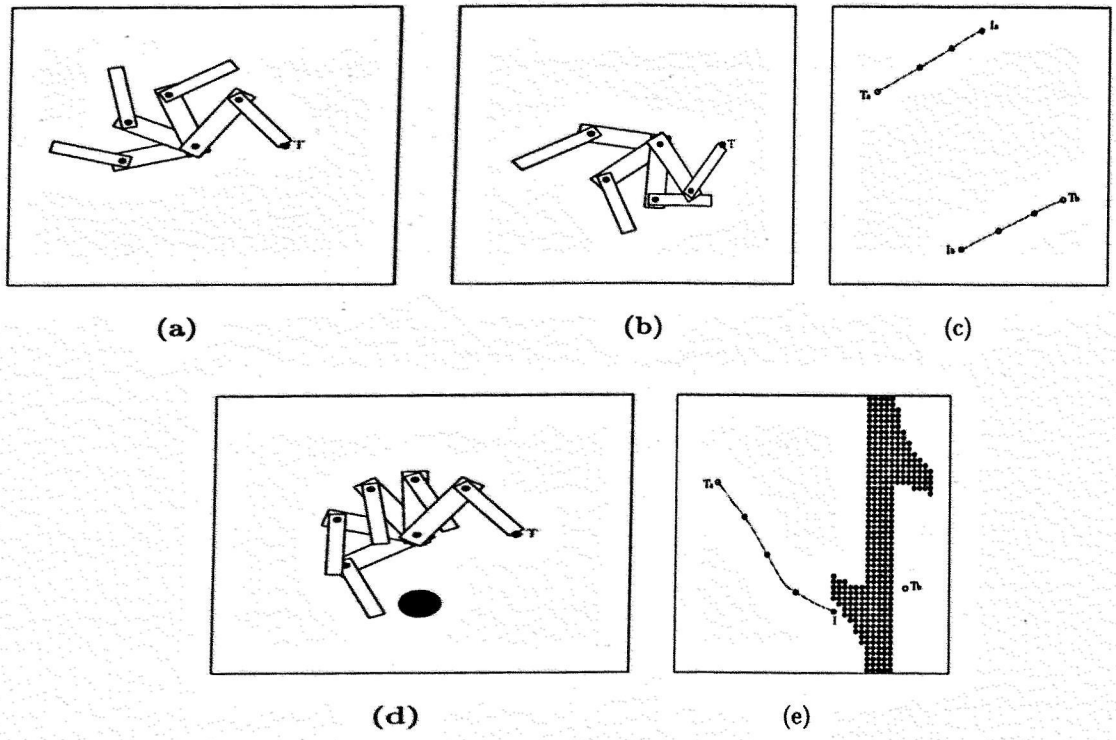


Figure 4.7: Manipulator configurations at four time instances during the movement to the command manipulator endpoint position (black dot, marked with a T). a), b) and c) when no obstacle are present. d) and e) in the case of one round obstacle in work space. a) Starting configuration and all intermediate configurations belong to one basin of attraction, leading to the first target configuration. b) Starting configuration and all intermediate configurations belong to the other basin of attraction, leading to the second target configuration. c) Paths in configuration space corresponding to the two manipulator movements resulting in the same manipulator end-point position. d) An obstacle (filled black circle) blocks one path. e) Representation of configuration space. Black dots are obstacle positions in configuration space corresponding to the obstacle in work space. The positions of the open circles correspond to the five manipulator configurations in (d). Note that the starting configuration (I) belongs to the first basin of attraction although the second target configuration is much closer in configuration space.

The first layer, the sensory map, consists of a large number of locally connected neurons. When the layer receives external input the activity pattern in the map starts to evolve towards a landscape with hills and ravines. The location of the tops of the hills in the map topographically coincide with target locations in state space, the bottoms of the ravines coincide with obstacle positions. Each path is determined by the neural activity gradient on the activity landscape in the sensory map.

The sensory map is functionally equivalent to the wave propagation or the distance transform model [33, 13, 24] and, therefore, it is able to find the shortest possible path. However, the previously mentioned models need additional modules to find the next node on the lattice with the largest activity to generate a smooth path along the discrete via-points.

The second layer, the motor map, is similar to the first layer but has in addition long range lateral connections and different inputs. During time evolution a cluster of activity shifts over the map directed by the input from the sensory map, until it reaches the target location. The movement of the cluster is a smooth path corresponding to a smooth motion in work space.

The motor map is functionally similar to the model suggested by Glasius et al. (1994) and by Droulez and Berthoz (1991). Compared to these models and compared to the potential field method [36, 42], the present model does not suffer from undesired local minima, e.g. due to concave shaped obstacles and is able to react to unforeseen external forces.

Some points are worth noting about the functional properties of the network.

A) The model is capable to "chose" one out of multiple targets. We can regard the set of initial positions which will lead to the same top as the basin of attraction belonging to that top.

B) In the case of a multi-link manipulator, more than one configuration may correspond to the same effector endpoint in work space. The model will make a smooth path from the present configuration to the closest target configuration. If an obstacle obstructs the path to the closest target in configuration space, then the basin of attraction belonging to the closest top is reduced in size while that belonging to the other tops has been enlarged. The model will choose the next nearest target configuration which is not obstructed.

C) The output of the motor map can be regarded as a motor command. In the case of a spring-like muscle system, each manipulator configuration can be accomplished by defining a particular set of spring constants [7, 29] or rest lengths [16] of the spring-like muscles. If each motor neuron k sends a motor

command proportional to $\frac{\theta_k(t)\,\sigma_k(t)\,\tau_k(t)}{\sum_k \sigma_k(t)\,\tau_k(t)}$, then the command manipulator position is a motor command consisting of votes of individual neurons to move the actuator to a certain position.

D) External forces acting on the subject, e.g. robot or manipulator, in work space may exist. Our model is capable to react to these forces and if the actuators are strong enough to overcome the external forces, and if the targets do not move too fast, a target configuration will be reached.

E) The inverse kinematics problem and the inverse dynamics problem, connected with the redundant manipulator, are problems distinct from the problem of trajectory formation discussed in this paper. The mapping from low-dimensional workspace to high-dimensional configuration space could be done with an additional layer. Each target and obstacle position in workspace correspond to a fixed subset of target and obstacle neurons in the neural map that depend on the shape of the manipulator only. If the actuators are strong enough, coriolis forces and inertia can not prevent the system to reach a target.

F) Like in other methods that use grids [47, 66, 36, 42, 53, 58, 75, 3, 11, 13, 5, 23, 24], the number of units grows exponentially with the number of degrees-of-freedom. Unlike the fully connected neural networks the number of connections in our model can be of the order of the number of units. One can think of different architectured or graded grids to circumvent the unit-consuming problem. In some 6DOF manipulators, the rotation of the gripper could be done without ever colliding into an obstacle. Hence the sensory map has to represent the targets only in this lower dimensional map and rotating of the gripper can be done in an one-dimensional sensory sub-map. This structure needs a smaller number of neurons but possesses the same functionality. Because the motor map produces continuous paths, not depending on the number of neurons in it, the number of neurons could be lowered to a minimum just enough to contain the area defined by $\tilde{r}$.

G) Pilot computer experiments demonstrate that the network performance is not sensitive to the choice of the transfer functions $g(x)$ and the neighborhood functions $f(x)$.

H) The model could be used with very simple connections which makes it possible to implement it into a fast analog electronic circuit. This can be used for autonomous robot trajectory formation in real time.

Chapter 5

The Population Vector, an unbiased estimator for non-uniformly distributed neural maps

Abstract

The Population Vector as a measure for the interpretation of neuronal activity in terms of sensory or motor events has been reliably used in the past for the case of uniformly distributed neuronal maps. In this study we will address the problem of the interpretation of neuronal activity in terms of the Population Vector for non-uniformly distributed maps. Based on mathematical analyses and on numerical computer simulations we demonstrate that, under some assumptions, the Population Vector also provides a proper estimate for neural maps with non-uniformly distributed neuronal response properties (i.e. the bias in the estimate, if present, is small). The main assumption is that the size of the receptive fields is not constant but that it is related to the density of receptive fields properties. The confidence level of the results is expressed by the variance of the estimate in the limit of a large number of neurons.

Adapted from: R. Glasius, A. Komoda and C.C.A.M. Gielen, The Population Vector, an unbiased estimator for non-uniformly distributed neural maps, submitted to *Neural Networks* 1996.

5.1 Introduction

In general, sensory and motor activity in biological neural networks is encoded by the activity of a large number of neurons. In order to provide insight in the representation of neuronal activity in motor cortex in monkey, Georgopoulos and collaborators [18, 19, 20] have formulated the hypothesis that the interpretation of motor activity in terms of planned movement direction could be estimated by a summation of the preferred directions of all neurons in an ensemble weighted by the firing rate of each single neuron. This weighted summation was defined as the Population Vector. A very similar approach was followed by Gielen et al. [21] to interpret the neuronal activity in the cat auditory nerve. In their approach they simply replaced each action potential by the linear estimate of the impulse response of each neuron and summated the results for all neurons. The result of this approach was a population vector, which provided an estimate for the auditory stimulus, that caused the activity in the auditory nerve.

In addition to the Population Vector other approaches have been proposed (e.g. [1, 6]) to interpret neuronal activity. A frequently used method is the Maximum Likelihood Estimator which theoretically provides the optimal result. In order to compare the performance of the Maximum Likelihood Estimator and the Population Vector, Seung and Sompolinsky [68] considered the theoretical case of a large set of neurons in visual cortex, each with a preferred direction, and studied the dependence of the performance on the variation of the size of the receptive fields of the neurons. In summary, they found that the Maximum Likelihood Estimator gave the best performance. However, the Population Vector appeared to be useful for several reasons, one of them being that the Population Vector is less sensitive to variations in the shape of the receptive field.

Most studies mentioned above explicitly or implicitly assumed a uniform distribution of preferred directions (or more generally, a uniform distribution of receptive field properties). However, non-uniform distributions of receptive field properties and preferred directions are a rule, rather than an exception in biological neural networks (see, for example, the receptive fields in superior colliculus [72, 62] and in visual cortex [32]). As a result, direct application of the Population Vector to non-uniformly distributed neuronal maps may fail to give a proper estimate of the sensory or motor events, which are encoded in the neural activity.

In another study Salinas and Abbott [64] proposed a linear reconstruction method, called the Optimal Linear Estimator, and tested this method on a set of neurons in visual cortex in order to reconstruct the stimulus orientation from measured firing rates. The latter method is based on a decorrelation of neuronal responses followed by weighted summation of the decorrelated responses.

This method gives a faster convergence than the usual Population Vector. The decorrelation procedure, however, requires non-local operations in order to calculate the proper weight factor for the contribution of each cell, which makes it a biologically implausible method.

In this article we study the concept of a Population Vector for the case of non-uniformly distributed maps, and we demonstrate how an unbiased estimate of the interpretation of neuronal activity can be obtained under some mild assumptions. We start with some theoretical analyses, addressing the problem for non-uniformly distributed maps, to find an expression for the expectation value and its variance. We show that the Population Vector estimate in general is biased but that the bias is small or absent provided that a few assumptions are met. The Population Vector can be used to interpret neuronal activity, since the assumptions, which make the Population Vector applicable to non-uniformly distributed maps, are approximately met by most neural maps known in sensorimotor pathways. The theoretical results are illustrated by numerical simulations.

5.2 Theory

Similarly to the earlier paper by Seung and Sompolinsky [68] we will investigate the firing frequencies $r_1, \ldots, r_N$ of N simplified neurons in response to a stimulus θ_s. These neurons can be thought to be the so-called simple cells in area V1 of the visual cortex, which have an orientation sensitive receptive field [32]. Let us assume that $\theta_s \in [-\pi, \pi]$ is the orientation of a visual bar-like stimulus. The responses r_k of neurons are modeled by independent random variables with a Poisson probability distribution of the form

$$P(r_k; \theta_s) = \frac{f(\theta_s - \theta_k, a)^{r_k}}{r_k!} e^{-f(\theta_s - \theta_k, a)} \tag{5.1}$$

where $f(\theta_s - \theta_k, a) > 0$ is the mean response $\langle r_k \rangle$ of neuron k, θ_k is the preferred direction of neuron k and a is the width of the receptive field of a neuron. The mean firing frequency function is expressed by

$$f(\theta_s - \theta_k, a) = \begin{cases} f_{min} + (f_{max} - f_{min}) \cos^2\left(\frac{\pi}{a}(\theta_s - \theta_k)\right) & \text{if } |\theta_s - \theta_k| < a/2 \\ f_{min} & \text{otherwise.} \end{cases} \tag{5.2}$$

Each neuron has a preferred direction $\theta_k \in [-\pi, \pi]$. When the stimulus θ_s is close to the preferred direction of the neuron, the probability of a large response r_k is

high. If the stimulus is outside the receptive field of the neuron, the response is small with a mean firing rate f_{min}.

We shall investigate the accuracy of the Population Vector estimate θ^* of an unknown stimulus θ_s. Unlike [68] we will consider non-uniform distributions of neurons assuming that the preferred directions θ_k of the neurons are distributed with a probability density $g(\theta)$. We chose for $g(\theta)$ a Gaussian distribution truncated to $[-\pi, \pi]$.

$$g(\theta) = \frac{\exp\left(\frac{-\theta^2}{2s^2}\right)}{\sqrt{2\pi s^2}\,\mathrm{erf}\left(\frac{\pi}{\sqrt{2}s}\right)} \tag{5.3}$$

Here, s is a real positive number and the Error function is defined as $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \exp(-t^2)dt$. For large values of s the distribution approximates a uniform distribution, while for small values of s the density becomes sharply peaked for preferred directions near zero.

We use the complex number representation $z = e^{i\theta}$ for a two-dimensional vector $(\cos\theta, \sin\theta)$. Keeping this in mind we define the Population Vector estimate as the weighted sum [68] of preferred directions

$$z^* = \frac{1}{N\alpha}\sum_{k=1}^{N} r_k e^{i\theta_k} \tag{5.4}$$

where $\alpha \in R$ is a normalization factor chosen in such a way that z^* becomes a unit vector in the limit of large N. The estimate z^* is an unbiased estimate if its expectation value $\langle z^* \rangle$ is equal to $e^{i\theta_s}$, where $< \ldots >$ represents the average with respect to the Poisson distribution (5.1).

In Appendix 1 we have derived the following expression for the expectation value $\langle z^* \rangle$ for large N

$$\begin{aligned}
\langle z^*(\theta_s)\rangle &= \frac{e^{i\theta_s}}{\alpha}\int_{-\pi}^{\pi} f(\phi, a)\, e^{i\phi} g(\phi + \theta_s)\, d\phi \\
&= \frac{e^{i\theta_s}}{\alpha}\int_{-\pi}^{\pi} f(\phi, a)\cos(\phi) g(\phi + \theta_s)\, d\phi \\
&\quad + i\frac{e^{i\theta_s}}{\alpha}\int_{-\pi}^{\pi} f(\phi, a)\sin(\phi) g(\phi + \theta_s)\, d\phi
\end{aligned} \tag{5.5}$$

The expression for the estimated angle θ^* is given by

$$\theta^* = -i \ln \frac{\langle z^* \rangle}{|\langle z^* \rangle|}$$

Since the first integral at the right hand side of equation (5.5) is real-valued, the first term in equation (5.5) has always the same direction as the optimal estimate $\langle z^* \rangle = e^{i\theta_s}$. The second term is always orthogonal to the optimal estimate and, therefore, it represents the bias of the estimate.

For a uniform distribution of preferred directions the function $g(\phi)$ is constant ($g(\phi) = (2\pi)^{-1}$), which makes the second term in equation (5.5) equal to zero because of the anti-symmetry of the sinus-function. In this case the normalization factor α reduces to $\int_{-\pi}^{\pi} f(\phi, a)\cos(\phi)d\phi$ which is equal to the first Fourier transform $\tilde{f}_1$. This implies that an unbiased estimate is obtained for a uniform distribution of preferred directions.

For non-uniform distributions, the second term in equation (5.5) will be different from zero in general. The relative magnitude of the first and second terms determines the effect of the bias in the estimate of the angle.

The considerations above show that the Population Vector estimator (5.4) applied directly to non-uniformly distributed maps is in general a biased estimator. We shall show, however, that the Population Vector defined by (5.4) can still be unbiased or , at least, that the bias will be sufficiently small, under some specific conditions.

The basic idea is that the bias will be small or zero, when the bias due to a changing density of preferred directions is compensated by a change in receptive field size. In detail, a bias induced by a larger number of neurons at one side of a stimulus due to a non-uniform distribution, could be compensated by reducing the size of the receptive field of these neurons, such that the number of neurons contributing to the Population Vector in the direction of higher density decreases with the result that the weighted summation of preferred directions is close to the stimulus value.

Therefore, we shall investigate the population vector for various combinations of distributions of preferred directions and of receptive receptive field sizes a. We shall demonstrate that for a particular choice of the function $a(\theta)$, namely when the receptive field size decreases for higher densities of preferred directions, the estimate θ^* deviates only slightly from the stimulus θ_s.

To find the optimal function $a(\theta)$ we define the following cost function:

$$C[a(\theta)] = \int_{-\pi}^{\pi} (\theta^*(a(\theta)) - \theta_s)^2 d\theta_s \qquad (5.6)$$

where θ^* is the Population Vector estimate for stimulus θ_s. The cost function is non-negative and for the unbiased estimate it has a minimal value equal to zero.

Our second goal is to find an expression for the standard deviation in the mean of the Population Vector estimate, which is represented by the variance of the estimate θ^*. An expression for this variance is derived in Appendix 2 (equation (5.9)) and is given by

$$\sigma_\theta^2 = \frac{1}{N\alpha^2}\int_{-\pi}^{\pi} f(\phi, a)\sin^2(\phi)g(\phi+\theta_s)d\phi$$

For a uniform distribution $g(\phi) = (2\pi)^{-1}$ the variance reduces to

$$\sigma_\theta^2 = \frac{1}{N\tilde{f}_1^2}\int_{-\pi}^{\pi} f(\phi, a)\left[\frac{1}{2} - \frac{1}{2}\cos(2\phi)\right]\frac{d\phi}{2\pi} = \frac{\tilde{f}_0 - \tilde{f}_2}{2N\tilde{f}_1^2}$$

as derived earlier in [68]. Here $\tilde{f}_n \equiv \int_{-\pi}^{\pi} f(\phi, a)\, e^{-in\phi}d\phi$ is the n-th order Fourier transform of f.

5.3 Numerical Simulations

In the numerical simulations we consider a sample of $N = 1000$ neurons with preferred directions drawn from the interval $[-\pi, \pi]$ with the probability density given by equation (5.3). For a finite number of neurons N the expectation value for z^* and the variance σ_θ^2 are given by

$$\begin{aligned}\langle z^*(\theta_s)\rangle &= \frac{1}{N\alpha}\sum_{k=1}^{N} f(\theta_k - \theta_s; a(\theta_k))e^{i\theta_k} \\ \sigma_\theta^2 &= \frac{1}{N^2\alpha^2}\sum_{k=1}^{N} f(\theta_k - \theta_s; a(\theta_k))\sin^2(\theta_k - \theta_s)\end{aligned}$$

where

$$\alpha = \frac{1}{N}\sum_{k=1}^{N} f(\theta_k; a(\theta_k))\cos(\theta_k)$$

Note that this normalization factor gives a unit vector when the bias is zero, i.e. in the case of a uniform distribution of preferred directions, or when $\theta_s = 0$. The orientation of the estimate does not depend on the normalization factor α. In our simulations we use $f_{min} = 10$ and $f_{max} = 1000$.

The task now is to find a function $a(\theta)$ which minimizes the cost function (5.6).

A search procedure for a function $a(\theta)$ without any constraints is a difficult variational problem since it requires to look for a minimum of the cost function

in the space of all functions for which the integral exists. Therefore, we restrict our search to functions which are polynomials in θ_k. Because $a(\theta_k)$ has to be larger than zero and because the distribution (5.3) is symmetric around zero we choose

$$a(\theta_k) = \sum_{l=0}^{L} c_l \, |\theta_k|^l$$

Neurons with preferred direction θ_k and $-\theta_k$ should have the same width of receptive fields because of the symmetric shape of $g(\theta)$. We use a set of stimuli which are uniformly distributed between -3.1 and 3.1 with step size 0.1 to find $a(\theta)$ which minimizes the cost function averaged over all stimuli

$$\theta_{s_i} \in \{-3.1, -3.0, \ldots, 3.0, 3.1\}$$

Our problem now reduces to minimization of the cost function

$$C(\{c_l\}) = \sum_i (\theta^*(\theta_{s_i}) - \theta_{s_i})^2 \tag{5.7}$$

with L unknown coefficients c_l. We look for the unknown coefficients c_l in the L-dimensional cube $[-1, 1]^L$ using a method of simulated annealing [38]. This optimization technique is well accepted to find the minimum in an energy landscape as defined by equation (5.7). It provides the opportunity to escape from local minima because steps to higher values of the energy function are allowed with a finite probability.

5.3.1 The uniform distribution

Figure 1 presents the mean of the variance over 10^3 stimuli in the estimate $z^*(\theta)$ as a function of the receptive field width a for two values for the number of neurons ($N = 10^3$, 10^4). Clearly there is an optimum for values near 0.8 radian. This value for a is similar to the result obtained in previous studies [68, 21] and corresponds to the receptive field width which gives the optimum in the amount of information in the neural activity.

This result can be understood from the following. For values of a near zero each neuron responds very selectively and many stimuli will not elicit any stimulus related neuronal activity at all. As a consequence the number of neurons responding to stimuli is small and the variance in the Population Vector due to the stochastic nature of the actionpotentials is large for very small values of a. For large values of a almost all neurons will respond to almost any stimulus.

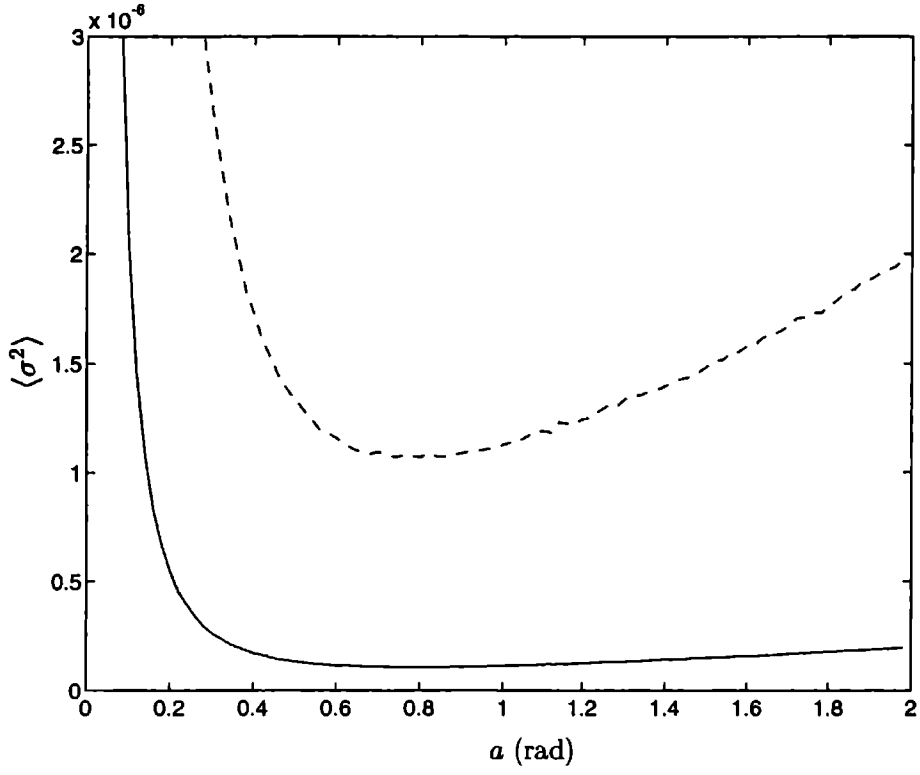


Figure 5.1: The mean variance $\langle \sigma^2 \rangle$ of the Population Vector estimate (averaged over 10^3 trials) as a function of the receptive field width a for $N = 10^3$ (dashed line) and $N = 10^4$ (solid line) for homogeneously distributed preferred directions of the neurons.

Hence, the accuracy to estimate the proper stimulus will be low and the variance will be large.

A larger number of neurons gives a smaller variance corresponding to a better estimate of the stimulus. The minimum in the variance shifts to smaller values of a for a larger number of neurons.

5.3.2 The non-uniform distribution

The results for a non-uniform distribution are shown in Figure 2. In this simulation, the Gaussian distribution of preferred directions, as defined in equation (5.3), has a standard deviation s equal to one radian. If we minimize the cost

function (5.7) by using the search algorithm outlined before, we find the set of receptive field widths $\{a_k\}$ shown in Figure 2a. The optimal values for a_k in the range between -2 and $+2$ radian vary by 1.2 radian and increase rapidly outside this range. Values for a receptive field width larger than ten are not shown. The largest receptive width was found for preferred directions near π and $-\pi$ and appeared to be as large as 1800. Figure 2b (solid line) shows the bias when the receptive fields are not constant but change as shown in Figure 2a. For comparison, we included the results when a constant value for a, which was optimal for the whole set of neurons, was chosen for all neurons (dashed line). For the former case (receptive field size related to density of preferred directions) the bias is much smaller over the whole range. However, the bias is not zero and the estimate can be missed by 0.2 radian. The corresponding variance in the estimate of the Population Vector is shown in Figure 2c.

The oscillations in Figure 2b are due to variations in the number of contributing neurons with a relatively broad tuning in the regions $[-2, -1.5]$ and $[1.5, 2]$. Therefore, and because of the low density of neurons in the outer regions, these oscillations can be considered to be a boundary artifact. In order to exclude these artifacts we have adjusted the cost criterion (5.7) in such way that we look for the distribution of receptive field widths which minimize the bias in the estimate, only in the region which contains 95% of the total number of neurons. For the Gaussian distribution with standard deviation $s = 1$, the stimulus region is $[-2, 2]$ rad. Hence, we took the sum in (5.7) over this segment for $\theta_{s_t} \in \{-2.0, -1.9, \ldots, 1.9, 2.0 \text{ (rad)}\}$. Note that we allow all neurons to contribute to the estimate for the proper stimulus.

Figure 3 shows the results of this analysis for the truncated cost criterion for the same Gaussian distribution ($s = 1$ rad). We have stopped the search procedure when the estimates differ less than 0.0075 radian from the true value, i.e. when $\forall_{\theta_{s_t}} : |\theta^*(\theta_{s_t}) - \theta_{s_t}| < 0.0075$. Figure 3a shows the optimal distribution of receptive field widths. Figure 3b (solid line) shows that with respect to the result in Figure 2b the new distribution shown in Figure 3a gives a bias which is significantly reduced in the interval $[-2, 2]$ radian. The dashed line in Figure 3b shows the bias for the case with the same (optimal) receptive field width for all neurons, as in Figure 2b. The variance in the range from -2 to $+2$ radian is small (Figure 3d) and is comparable to the variance shown in Figure 2c. Figure 3c illustrates that the optimization procedure gives receptive field widths which are smaller for higher densities.

To illustrate the fact that contributions of neurons with a preferred direction far from the stimulus value can contribute to a reduction of the bias we present

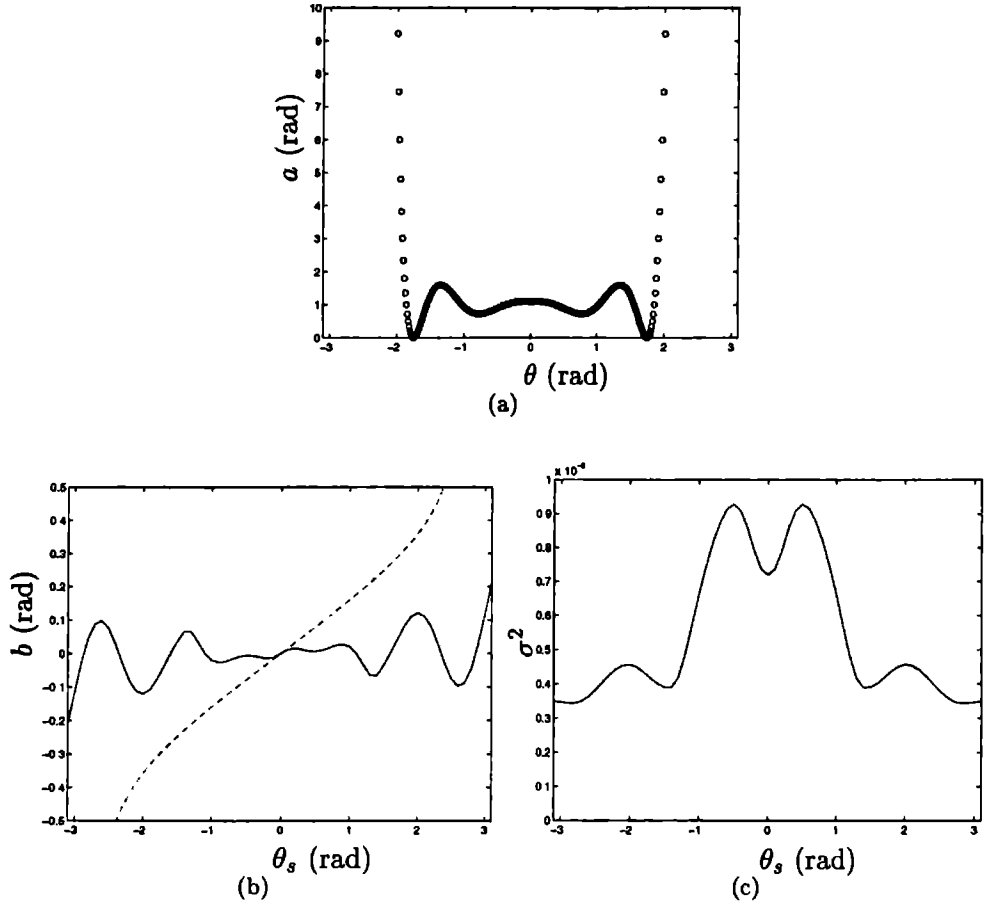


Figure 5.2: The results of a numerical simulation with distribution $g(\theta)$ and $s = 1.0$ and $N = 1000$. a) The optimal width a for neurons with various preferred directions. For each neuron the combination (θ_k, a_k) is indicated with an open dot. Receptive field widths larger than ten are not displayed. b) The bias $b(\theta^*)$ in the estimate as a function of the stimulus value θ_s (solid line). In the case of an optimal constant receptive field width for all neurons ,i.e. $a = 1.1$ (dashed line). c) The variance of the Population Vector for stimulus values θ_s.

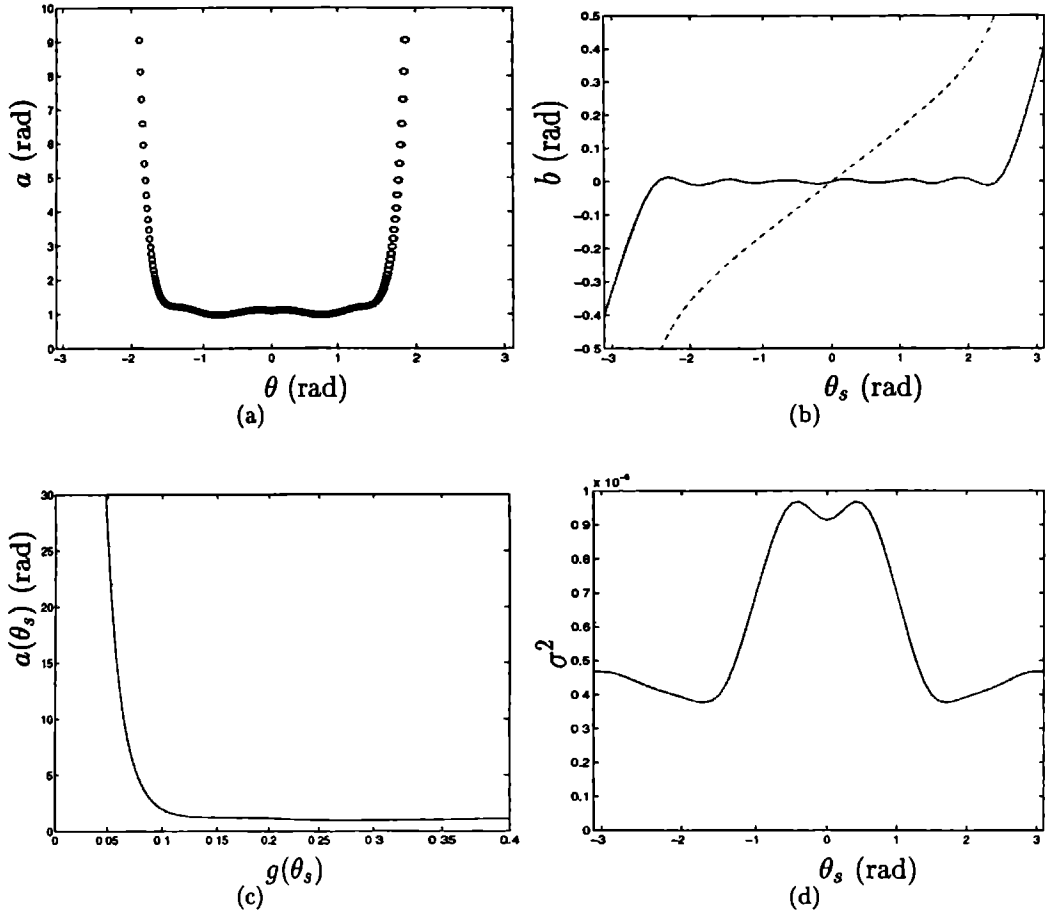


Figure 5.3: The results of the numerical simulations with distribution $g(\theta)$ with $s = 1.0$ (rad), $N = 1000$. The cost is integrated over the interval $\theta_s \in [-2.0, 2.0]$. a) The optimal width a for neurons as a function of preferred direction. For each neuron the combination (θ_k, a_k) is indicated with an open dot. Receptive field widths larger than ten are not displayed. b) The bias $b(\theta^*)$ in the estimate as a function of the stimulus value θ_s (solid line) and for the case of an optimal constant receptive field width for all neurons ($a = 1.1$, dashed line). We have stopped the search procedure when each estimate differs less than 0.0075 radian from the true value, i.e. when $\forall_{\theta_{s_i}} : |\theta^*(\theta_{s_i}) - \theta_{s_i}| < 0.0075$. c) The optimal receptive field width $a(\theta)$ as a function of the density $g(\theta)$. d) The variance of the Population Vector for various stimulus values θ_s.

the results of Figure 3 in an alternative way in Figure 4. In Figure 4a we have calculated the mean firing frequency for some stimuli $\theta_s \in [0, \pi]$ and for all neurons. Figure 4b is a cross section of Figure 4a, where the mean firing frequency of all neurons is shown for the case $\theta_s = 0$. The graph is symmetric. Near the stimulus value $\theta_s = 0$ a group of neurons responds and "votes" for an estimate near zero. However, also in the far regions near π and $-\pi$ some neurons with large receptive field widths respond to the stimulus. Because of the symmetry however, the total contribution to the estimate is zero. Hence an unbiased estimate $\theta^* = 0$ is the result.

When the stimulus orientation is $\theta_s = 0.2$, there is a cluster of responding neurons with preferred directions near $\theta_s = 0.2$ and a group of neurons with preferred directions near π and $-\pi$ (Fig. 4c). Note that the cluster of neurons with a preferred direction near $\theta_s = 0.2$ is asymmetric with a larger number of responding neurons with preferred directions near zero due to the larger density of preferred directions near $\theta_s = 0$. This asymmetry causes a bias in the population vector, which is offset by the somewhat larger group of responding neurons with a preferred direction near π. As a result, the bias in the Population Vector is small (Figure 3b, solid line). For a stimulus value $\theta_s = 1$ (Figure 4d) the two groups of responding neurons overlap.

The occurrence of two disconnected groups of responding neurons does not agree with experimental data [72, 32, 62] which report only one group of responding neurons. In fact the large receptive field widths near the extremes $-\pi$ and π are an artifactual boundary effect. Therefore, additional constraints have to be formulated in order to find results which will meet the experimental results. We regard the group of neurons with a preferred direction equal to or close to the stimulus value as the main group. Neurons do not belong to the main group if their response $f(\theta_s - \theta_k, a_k)$ is larger than the response of the neighboring neuron which has a preferred direction closer to the stimulus. We replace the mean firing frequencies of all the neurons not belonging to the main group by the minimal mean firing frequency value f_{min}. The width of the receptive field a has been adjusted so as to minimize the bias.

Figure 5 shows the result of the constrained search algorithm. The bias in the estimate θ^* is slightly larger in Figure 5b than in Figure 3b. Figure 5c shows that the optimal distribution of receptive fields corresponds to smaller receptive field widths for higher densities of neurons. Due to the smaller receptive field width of neurons at higher densities the number of neurons with a preferred direction smaller and larger than the stimulus value is approximately constant giving rise to the relatively small bias in Figure 5b. The variance in the population vector

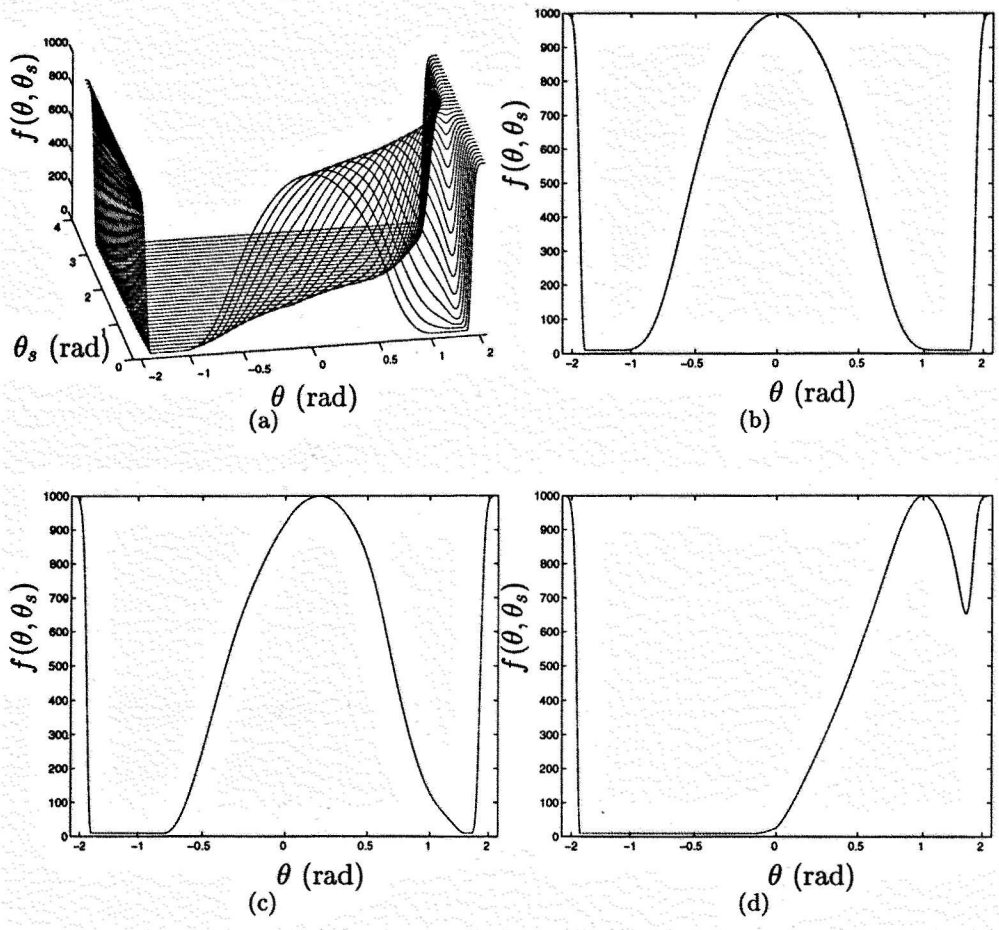


Figure 5.4: a) The mean firing frequency of neurons versus the preferred direction of each neurons for some stimuli θ_s. $N = 1000$ and $s = 1.0$. b) cross section of (a) for $\theta_s = 0$. c) cross section of (a) for $\theta_s = 0.2$. d) cross section of (a) for $\theta_s = 1$. Note that the θ-axis is scaled in a way such that the density of neurons along the axis becomes uniform.

estimate is slightly larger in Figure 5d than in Figure 3d.

The firing rate of the neurons responding to various stimuli is shown in Figure 6. For all stimuli a single cluster of neurons is responding. Notice that the cluster

of responding neurons is smaller than in Figure 5. This is related to the fact that the width of the receptive fields for this model in the range of preferred directions $[-1, 1]$ is smaller than for the model shown in Figure 3.

5.3.3 Robustness for variations in receptive field width

Although there is a general tendency for neurons in the visual system and in the auditory system to have smaller receptive fields when the density of neurons is higher (in accordance with the result in Figure 5c) there is a considerable variability in receptive field width. Therefore, we have investigated the robustness of the population vector estimate for deviations of the optimal receptive field width as shown in Figure 5c. The results are shown in Figure 7.

Multiplication of all receptive field widths by a factor of 1.5 did not have a large effect on the bias (compare Figure 5b and Figure 7a). This illustrates that it is the relative size of receptive field widths which is important to minimize the bias in the population vector estimate, rather than absolute size of the receptive field width.

The population vector estimate is also robust for random variations in the receptive field width a. Inducing a variability in receptive field width by multiplication of each a-value by the uniformly distributed random variable z in the interval $[0, 5]$ for each neuron has only a small effect on the bias (compare Figure 5b and Figure 7b).

Simulations in which the density of preferred directions was randomly modified by 25% also demonstrated that the bias and variance remained the same within 5% in the interval $[-2, 2]$.

5.4 Discussion

The Population Vector coding has been a popular way of interpreting neuronal activity. However, this method has frequently been associated with the assumption of homogeneous sampling of the input space by neurons. The main result of this study is that the population vector can also provide an accurate estimate of the sensori-motor event when the distribution of receptive field properties is not homogeneously distributed. In that case the population vector can provide an accurate result if there is an approximately inverse relation between the receptive field width and the density of receptive fields. Random variations in receptive

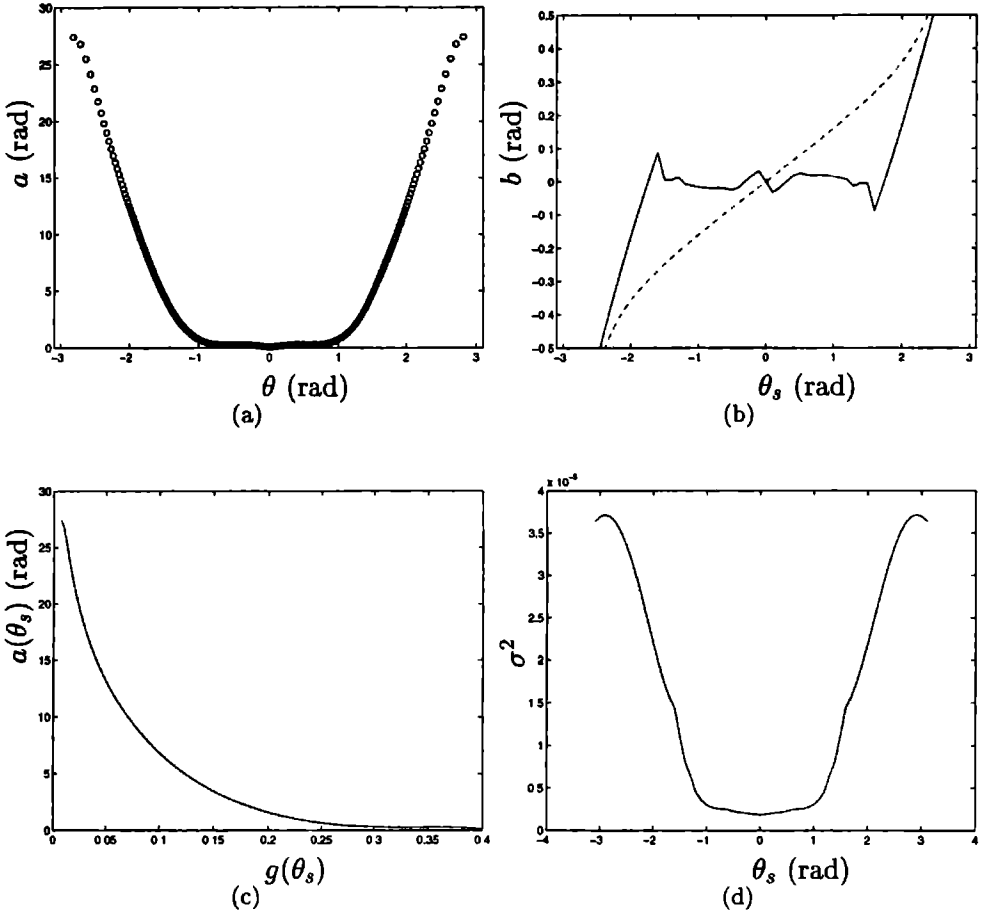


Figure 5.5: The results of a numerical simulation with distribution $g(\theta)$ and $s = 1.0$, $N = 1000$ and with the constraint of one single responding group of neurons. The cost is integrated over the interval $\theta_s \in [-2.0, 2.0]$. a) The optimal width a for neurons with various preferred directions. For each neuron the combination (θ_k, a_k) is indicated with an open dot. b) The bias $b(\theta^*)$ in the estimate as a function of the stimulus value θ_s (solid line). In the case of an optimal constant receptive field width for all neurons ,i.e. $a = 1.1$ the bias is indicated by the dashed line. c) The optimal receptive field width $a(\theta)$ as a function of the density $g(\theta)$. d) The variance of the Population Vector for various stimulus values θ_s.

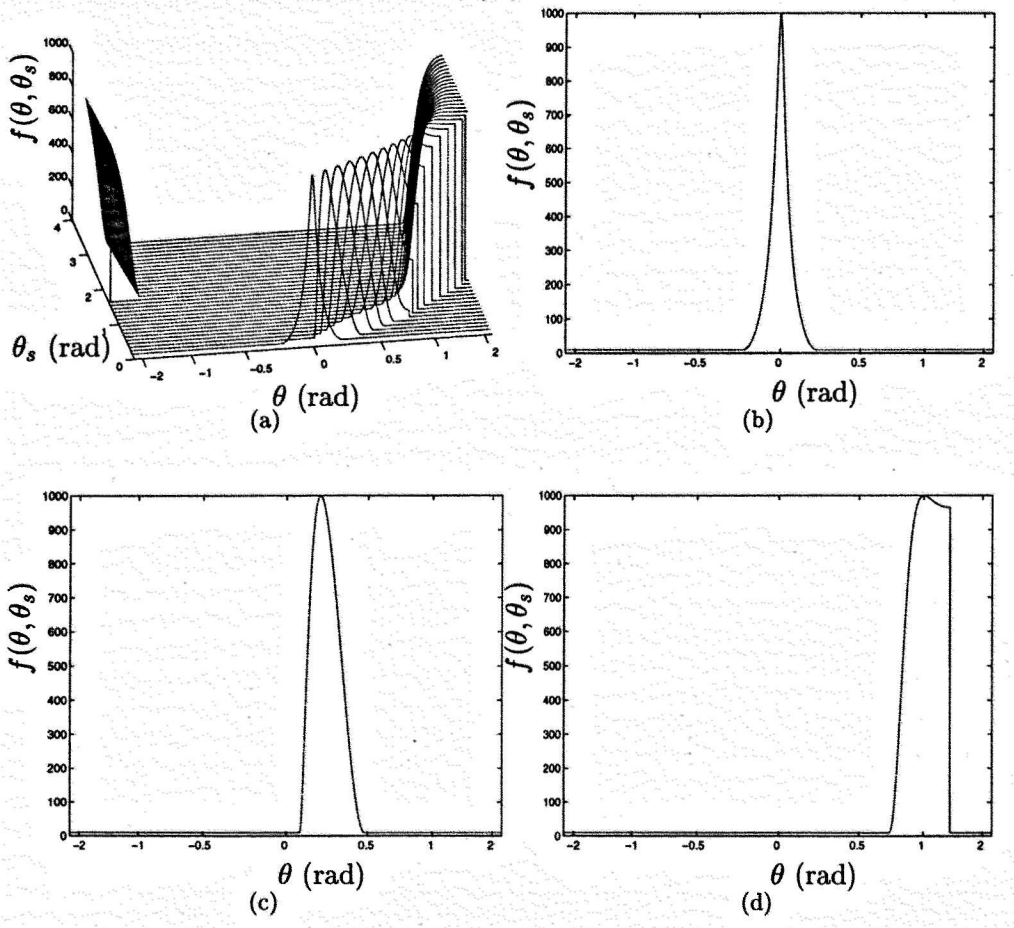


Figure 5.6: a) The mean firing frequency of the neurons versus the preferred direction of the neurons for some stimuli θ_s. $N = 1000$, $s = 1.0$ The constraint was imposed that only one group of neurons was responding for each stimulus. b) cross section of (a) for $\theta_s = 0$. c) cross section of (a) for $\theta_s = 0.2$. c) cross section of (a) for $\theta_s = 1$. Note that the θ-axis is scaled in a nonlinear way such that the density of neurons along that axis is uniform.

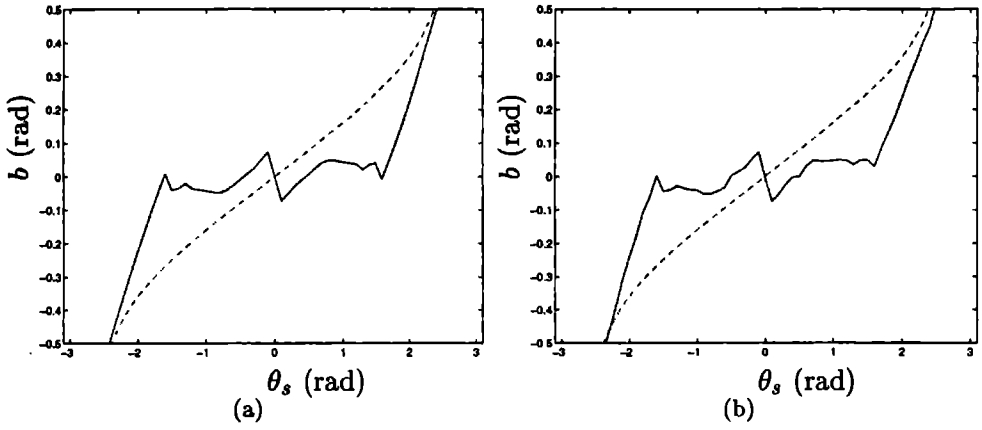


Figure 5.7: The bias $b(\theta^*)$ in the estimate as a function of the stimulus value θ_s (solid line), where $N = 1000$, $s = 1.0$, and with the constraint of one responding group for each stimulus. The dashed line represents the bias in the case of an optimal constant receptive field width for all neurons ($a = 1.1$). a) The parameter a has been multiplied by 1.5 for all neurons. b) The parameter a has been multiplied by z randomly chosen from the interval [0,5].

field properties do have a small effect only on the population vector estimate, presumably because the variations in the contribution of cells due to the variations in receptive field width disappear in the summation of the contributions of all responding neurons.

In this discussion we will first concentrate on the question, whether the assumptions, which underlie our method, are in agreement with experimental observations. After that, we will discuss the significance and relevance of the results of this paper in the context of previous publications.

In the visual system it is well known that the density of receptive fields is much higher near the fovea than it is towards the periphery. This has been found both in retina [54] and in visual cortex [55, 63, 65], where the density of receptive fields decreases exponentially towards the periphery. The fact that a larger part of visual cortex is devoted to the central visual field is summarized by the cortical magnification factor [63, 65, 15]. It is also a well known fact, that the mean receptive field size increases from the fovea towards the periphery in retina

[55] and visual cortex [15]. In cat retinal ganglion cells the receptive field size of the center is approximately inversely related to the density of receptive fields [55] In monkey visual cortex the receptive field size decreases with the inverse of the cortical magnification factor [15]. These literature data indicate that, qualitively, the receptive field size and density of receptive fields in retina and cortex are inversely related, as suggested by the results of this study.

Based on studies using electrical stimulation [61, 72] proposed a logarithmic function to describe the mapping from retinal coordinates to the coordinates in the superior colliculus. This logarithmic function implies that the density of neurons decreases exponentially towards the periphery, like in visual cortex. The receptive field size was assumed to be constant in collicular coordinates based on the findings of [50]. This implies that the receptive field size increases logarithmically in retinal coordinates (i.e. with the inverse of the exponential density function !). Due to this inverse relationship of the functions which describe density and receptive field size of neurons as a function of retinal eccentricity, the population estimate of collicular activity in [72] gave a reliable, unbiased estimate of target position in space.

In the auditory system, the distribution of fibers in the acoustic nerve is ordered as a function of frequency. The density of fibers is highest at the higher frequencies and decreases for lower frequencies [37]. The equivalent of the receptive field size for auditory nerve fibers, the width of the frequency tuning curve or the inverse of the sharpness of tuning, is smaller for nerve fibers tuned at high frequencies (i.e. when the density of fibers is high) and increases for lower frequencies [37]. The fact that the population vector could be used to provide a sensory interpretation of the stimulus-related activity in the acoustic nerve with its non-uniform distribution of tuning curves [22] was possible thanks to the inverse relation between density of neurons and tuning curve of neurons. These examples of previous studies, which have successfully used the population vector idea for cases of non-uniformly distributed receptive fields supports the theoretical ideas outlined in this study.

The problem that a non-uniform distribution of receptive fields might give rise to artifactual results for the Population Vector has been recognized before in other studies. [64] presented a method which corrects for the correlation in firing rate of neurons caused by the fact that the optimal stimulus is not orthogonal for different neurons. This method also corrects for non-uniformities in the distribution of neurons and in addition gives a better signal-to-noise-ratio than the well known Population Vector. This method is certainly a good way to interpret the neuronal activity. Whether or not this method is also of relevance for our understanding

of neuronal information processing in biological neural networks remains an open question, since this procedure requires non-local operations on neuronal activity information, which is in general assumed to be non-realistic from a biological point of view. Since motocortical cells are known to have monosynaptic contacts with motorneurons of distal muscles [57] the advantage of the population vector is that it bears more resemblance with the way in which activity in motor cortex becomes evident for external observation as the summation of motor-unit twitches, which give rise to force and movements.

The search procedure to find the optimal value for the parameter a (which corresponds to half the receptive field width) gives a value of about 1.1 radian (Figure 3a), which is close to the optimal value of 0.8 radian in Figure 1. However, it should be noted that the optimal a-value in Figure 1 is found by minimization of the variance σ^2, whereas the optimal a-values shown in Fig. 3a are found by minimization of the bias. Apparently, both procedures give similar results.

When the restriction is imposed that only one group of neurons responds to each stimulus, the optimal a-value appears to become somewhat smaller. We also found that multiplication of all receptive field widths by the same constant does hardly affect the bias (see Fig. 7a). This can be understood from the data in Fig. 1, which shows that the curvature in the minimum of the variance as a function of a is rather small. As a consequence, the same variations in a for all neurons will not have a large impact on the variance of the estimate. Therefore, different constraints on the model give rise to different values for the optimal a-value without large effects on the variance.

Random variations in the receptive field width hardly affect the population vector estimate either. This is mainly because any effects of the random variations in receptive field width disappear by adding the responses of many responding neurons. For stimulus values near π and $-\pi$ the averaging effect is smaller due to the smaller density of neurons. Hence, variations in a give rise to a larger bias and variance for smaller densities of neurons.

In summary, we conclude that the Population Vector can be used as an estimate for the sensory or motor interpretation of neural activity for non-uniform maps when the density of receptive field properties and receptive field are inversely related. The latter assumption seems to be in agreement with present knowledge about neuronal information processing in most sensory systems.

5.5 Appendix

5.5.1 The expectation value $\langle z^* \rangle$

In this appendix we derive an analytical expression for the expectation value of the Population Vector estimate for the case of non-uniformly distributed preferred directions.

$$\begin{aligned} \langle z^* \rangle &= \left\langle \frac{1}{Nn} \sum_{k=1}^{N} r_k e^{i\theta_k} \right\rangle \\ &= \frac{1}{Nn} \sum_{\theta_k=\theta_1}^{\theta_N} f\left(\theta_k - \theta_s, a\right) e^{i\theta_k} \end{aligned}$$

If N is large the sum can be replaced by an integral

$$\langle z^* \rangle = \frac{1}{\alpha} \int_{-\pi}^{\pi} f\left(\theta - \theta_s, a\right) e^{i\theta} g\left(\theta\right) d\theta$$

Changing variables $\phi = \theta - \theta_s$ and assuming a symmetric function of the density of preferred directions results into

$$\begin{aligned} \langle z^* \rangle &= \frac{e^{i\theta_s}}{\alpha} \int_{-\pi}^{\pi} f\left(\phi, a\right) e^{i\phi} g\left(\phi + \theta_s\right) d\phi \\ &= \frac{e^{i\theta_s}}{\alpha} \left(\int_{-\pi}^{\pi} f\left(\phi, a\right) \cos(\phi) g\left(\phi + \theta_s\right) d\phi + i \int_{-\pi}^{\pi} f\left(\phi, a\right) \sin(\phi) g\left(\phi + \theta_s\right) d\phi \right) \end{aligned}$$

5.5.2 Variance of the directional fluctuations

For large N z^*, as a sum of independent random variables, has a two dimensional Gaussian distribution. This distribution is completely determined the correlation matrix

$$C = \begin{pmatrix} \left\langle (x - \tilde{x})^2 \right\rangle & \left\langle (x - \tilde{x})(y - \tilde{y}) \right\rangle \\ \left\langle (x - \tilde{x})(y - \tilde{y}) \right\rangle & \left\langle (y - \tilde{y})^2 \right\rangle \end{pmatrix} \tag{5.8}$$

where $x = \mathrm{Re}\tilde{z}$ and $y = \mathrm{Im}\tilde{z}$.

To calculate the variance in z^* We define

$$\begin{cases} x \equiv \mathrm{Re}\left(z^*\right) = \frac{1}{Nn} \sum_{k=1}^{N} r_k \cos(\theta_k) \\ y \equiv \mathrm{Im}\left(z^*\right) = \frac{1}{Nn} \sum_{k=1}^{N} r_k \sin(\theta_k) \end{cases}$$

Using the shorthand notation $f_k = f(\theta_k - \theta_s, a)$, the variance in x is

$$\begin{aligned}
\left\langle (x-\bar{x})^2 \right\rangle &= \left\langle \left(\frac{\sum_{k=1}^{N} r_k \cos(\theta_k)}{Nn} - \frac{\sum_{k=1}^{N} f_k \cos(\theta_k)}{Nn} \right)^2 \right\rangle \\
&= \frac{1}{N^2\alpha^2} \left\langle \left(\sum_{k=1}^{N} (r_k - f_k) \cos(\theta_k) \right)^2 \right\rangle \\
&= \frac{1}{N^2\alpha^2} \left\langle \sum_{k=1}^{N} (r_k - f_k)^2 \cos(\theta_k)^2 \right. \\
&\quad \left. + \sum_{k \neq l}^{N} (r_k - f_k)(r_l - f_l) \cos(\theta_k)\cos(\theta_l) \right\rangle \\
&= \frac{1}{N^2\alpha^2} \left\langle \sum_{k=1}^{N} (r_k - f_k)^2 \cos(\theta_k)^2 \right\rangle \\
&= \frac{1}{N^2\alpha^2} \sum_{k=1}^{N} f_k \cos^2(\theta_k)
\end{aligned}$$

In the latter two steps we use the fact that $(r_k - f_k)$ are independent random variables and the fact that $< (r_k - f_k)^2 >= \sigma_{r_k}^2 = f_k$.

Likewise we calculated the variance in y and the covariance

$$\begin{aligned}
\left\langle (y-\bar{y})^2 \right\rangle &= \left\langle \left(\frac{\sum_{k=1}^{N} r_k \sin(\theta_k)}{Nn} - \frac{\sum_{k=1}^{N} f_k \sin(\theta_k)}{Nn} \right)^2 \right\rangle \\
&= \frac{1}{N^2\alpha^2} \sum_{k=1}^{N} f_k \sin^2(\theta_k)
\end{aligned}$$

$$\begin{aligned}
\langle (x-\bar{x})(y-\bar{y}) \rangle &= \left\langle \left(\frac{\sum_{k=1}^{N} r_k \cos(\theta_k)}{Nn} - \frac{\sum_{k=1}^{N} f_k \cos(\theta_k)}{Nn} \right) \right. \\
&\quad \left. \cdot \left(\frac{\sum_{k=1}^{N} r_k \sin(\theta_k)}{Nn} - \frac{\sum_{k=1}^{N} f_k \sin(\theta_k)}{Nn} \right) \right\rangle \\
&= \frac{1}{N^2\alpha^2} \sum_{k=1}^{N} f_k \cos(\theta_k) \sin(\theta_k)
\end{aligned}$$

The two eigenvalues of the matrix correspond to eigenvectors, in the direction of the orientation θ and in the direction of the magnitude $R = |z^*|$. Hence the

fluctuations in θ are given by the eigenvalue corresponding to the eigenvector in the direction θ. It is equal to

$$\begin{aligned} \sigma_\theta^2 &= \frac{1}{N\alpha^2}\int_{-\pi-\theta_s}^{\pi-\theta_s} f(\phi,a)\left[\frac{1}{2}-\frac{1}{2}\cos(2\phi)\right] g(\phi+\theta_s)d\phi \\ &= \frac{1}{N\alpha^2}\int_{-\pi-\theta_s}^{\pi-\theta_s} f(\phi,a)\sin^2(\phi) g(\phi+\theta_s)d\phi \end{aligned} \tag{5.9}$$

Chapter 6

Inleiding en Samenvatting

Ondanks dat er een enorme vooruitgang in de kunstmatige intelligentie is geboekt, kan het zich bij lange na niet vergelijken met natuurlijke intelligentie: het gedrag van biologische organismen in de diverse omstandigheden van hun natuurlijke biotoop. Vooral in een gebied vol ruis en wazige, slecht gedefinieerde omstandigheden, wat typerend is voor een natuurlijke omgeving, zijn de prestaties van kunstmatige systemen dramatisch slecht. In dit opzicht zijn mens en dier opmerkelijke wezens. Met slechts weinig moeite zijn zij in staat om een visueel beeld te analyseren, de interessante objecten te herkennen, een doel te kiezen, zichzelf te verplaatsen en zodoende binnen een complexe, dynamische omgeving zonder botsingen het doel te bereiken. Het is bijvoorbeeld verwonderlijk hoeveel mensen er in een warenhuis kunnen, voordat er botsingen plaats vinden.

Om de veranderingen in zijn omgeving op te merken bezit de mens een groot aantal sensoren waarvan elk een continue vloed van afferente neurale signalen verwekt. Deze signalen vormen een enorme stroom van informatie binnen het centrale zenuwstelsel. Neem bijvoorbeeld het lopen. Een enorme hoeveelheid sensorische neuronen in de ogen en in het evenwichtsorgaan als mede de receptoren in de huid en in de spieren projecteren naar specifieke centra in het centrale zenuwstelsel. Elk centrum bestaat uit een bijzonder groot aantal neuronen. De centra zijn door middel van een zeer complexe bedrading met elkaar verbonden met die gebieden welke met de coordinatie van de spieren te maken hebben. In dit voorbeeld is timing essentieel en hebben de informatiestromen een cyclisch karakter. Vanuit dit standpunt is zo'n eenvoudige taak als "lopen in een warenhuis" een meesterwerk van complexe procesbeheersing. Het centrale zenuwstelsel is zondermeer een zeer krachtig en ingewikkeld systeem dat reeds generaties van onderzoekers heeft geinspireerd om een beter begrip over zijn werking te verkrij-

gen en het zal nog generaties blijven inspireren.

Naast een beter begrip van de fundamentele processen in de onderliggende hersenfuncties binnen biologische systemen, kunnen de resultaten van wetenschappelijk onderzoek soms ook worden toegepast in kunstmatige systemen. Een voorbeeld hiervan is de ontwikkeling van neurale protheses die niet goed functionerende neurofysiologische systemen kunnen vervangen. De kunstmatige cochlea wordt bijvoorbeeld al veel gebruikt en de ontwikkeling van een kunstmatige retina maakt goede vooruitgang. Daarnaast kunnen de resultaten ook gebruikt worden om robotten en artificiele intelligentie aan menselijke eigenschappen te helpen.

In deze thesis zullen wij een aantal neurale netwerkmodellen beschrijven die gebruikt worden voor padplanning en het ontwijken van obstakels. De kern van het probleem is het zoeken naar een pad tussen de initiele positie en de doelpositie van een subject of van een robot. Het subject of de robot moet een pad vinden zodanig dat het doel wordt bereikt zonder bewegende of niet bewegende objecten te raken. Hierbij hoeft het resultaat niet per se het kortst mogelijke pad te zijn. Om obstakels te ontwijken zal de robot of subject het pad op enige afstand van de obstakels laten lopen. In het kortste pad zit noodzakelijkerwijs een aantal scherpe bochten om de obstakels heen hetgeen een niet zo efficiente manier van bewegen is. Eventueel zou er een kostenfunctie in het systeem gebouwd kunnen worden, waardoor het meer kosten met zich mee brengt om bijvoorbeeld via een rechte lijn naar het doel te bewegen over een bergrug dan via een langer pad langs de berg.

In de literatuur wordt naar dit probleem verwezen als zijnde "padplanning" of "padontwikkeling". Padplanning gaat er van uit dat het "waar" en het "wanneer" van de objecten in de omgeving van tevoren bekend zijn. De padplanner kan dan, rekening houdende met de posities van de objecten op de betreffende tijden, verschillende paden tussen de initiele positie en de doel positie berekenen. Bij padontwikkeling moet het systeem rekening houden met de continue veranderingen in de omgeving gedurende het planningsproces. Dit is een continu proces zodat op elk tijdstip, afhankelijk van de posities van de bewegende objecten, er steeds weer een nieuw pad berekend moet worden. Uiteindelijk is het de bedoeling dat, indien het mogelijk is, het (bewegende) doel bereikt wordt. Zowel padplanning als padontwikkeling spelen een belangrijke rol in veel toepassingen. Voorbeelden hiervan zijn het besturen van autonome robotten of robotarmen, het zoeken van een optimale routing op een chip of op een printplaat en het navigeren in het verkeer.

In de begindagen werden sequentiele algoritmen ontwikkeld die een pad voor een autonome robot binnen een statische omgeving konden berekenen [4, 12, 35, 41, 47, 76]. In het algemeen verdeelden deze algoritmen de vrije werkruimte in deelgebieden om vervolgens een boomstructuur op te zetten bestaande uit alle mogelijke verplaatsingen tussen de deelgebieden die uiteindelijk de initiele positie met de doelpositie verbinden. Vervolgens werd voor elke oplossing de lengte van het pad berekend. Het kortste pad werd uiteindelijk verkozen als de optimale oplossing. Deze brute-force methode leed aan een aantal ernstige tekortkomingen. Wanneer er een hogere resolutie van deelgebiedjes noodzakelijk was, ontstond een combinatorische explosie van mogelijke oplossingen die weer tot onacceptabel lange berekeningen leidde. Een bijkomend nadeel was dat de omgeving niet mocht veranderen, omdat de tijdvretende constructie van de boom en de extensieve zoekmethode niet real-time was te realiseren.

Vervolgens werd de "veldpotentiaal" methode ontwikkeld[36, 42], die daarentegen wel in een dynamische omgeving kon functioneren. De kern van deze methode berustte op het toewijzen van virtuele elektrische ladingen aan alle objecten. Het doel ontving een aantrekkende lading en de obstakels ontvingen een afstotende lading. Door gebruik te maken van de fysische wetten van de elektrodynamica, was het algoritme in staat om de krachten op en de daaruit voortvloeiende bewegingen van de robot te berekenen. Deze benadering genereerde een gelijkmatig en continu pad. Echter, ondanks enkele pogingen om het model aan te passen [3, 5, 11, 53, 75], was het mogelijk dat het algoritme in een lokaal minimum werd gevangen (zie figuur 6.1). De lokale minima waren het resultaat van de vaak complexe configuratie van de afstotende obstakels en het aantrekkende doel. In dergelijke gevallen bestond de enige oplossing om het doel te bereiken uit een verplaatsing van de robot in de richting van een oplopende potentiaalgradient.

In tegenstelling tot de "veldpotentiaal" methode kon de "golfvoortplanting" of de "afstand getransformeerde" methode [13, 47, 33, 58, 66] altijd de oplossing, als die bestond, vinden. In dit model was de gehele werkruimte onderverdeeld in kleine subruimten. Elke subruimte werd door een knoop in een netwerk gerepresenteerd waarbij aan elkaar verbonden knopen aan elkaar grenzende subruimten voorstelden. De knoop overeenkomende met de subruimte welke het doel bevatte, stuurde een signaal naar al zijn buren die geen obstakel in de bijbehorende subruimte hadden. In de volgende stap zonden deze buren op hun beurt een signaal naar hun obstakelvrije buren, enz. Elke keer dat de knoop een signaal naar zijn buren stuurde, werd zijn activatiewaarde met een verhoogd. De signalen verspreidden zich over de knopen als een lopende golf die telkens die knopen ontweek welke met obstakels overeenkwamen. Zo gauw de knoop die met de huidige posi-

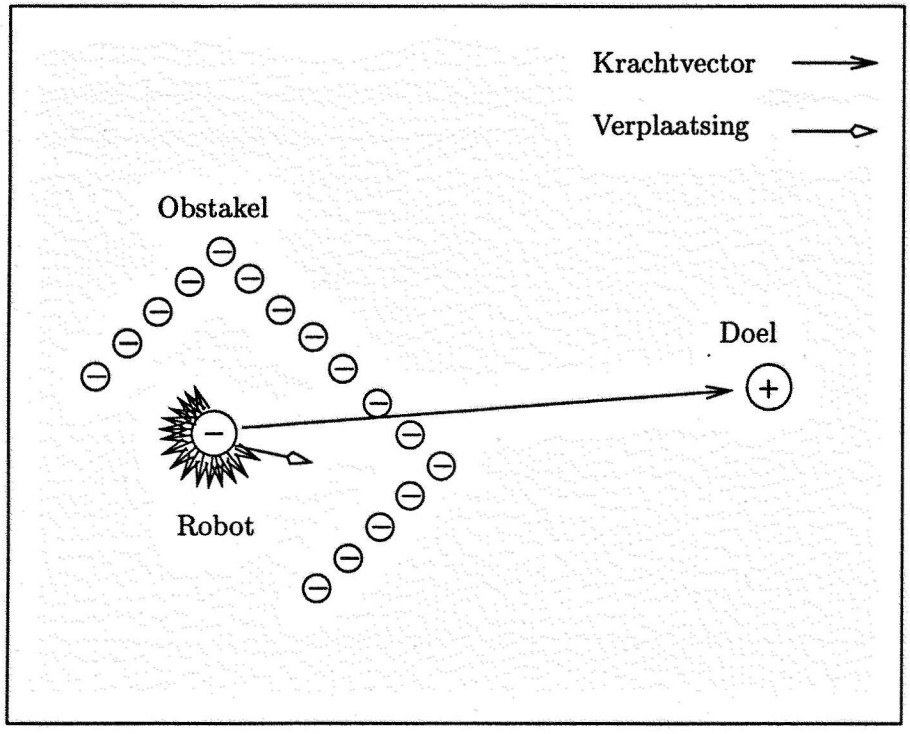


Figure 6.1: In de "veldpotentiaal" methode mocht alleen de robot volgens de wetten van de elektrodynamica bewegen. De obstakels kregen een afstotende negatieve lading en het doel een aantrekkende positieve lading. De verplaatsing van de robot was het resultaat van alle aantrekkende en afstotende krachten (pijlen) in het systeem. In dit geval zou de robot gevangen worden in de concave hindernis.

tie overeenkwam een signaal ontving, stopte het proces. Vervolgens werd vanuit die laatste knoop een ketting van opeenvolgende verplaatsingen gemaakt. Steeds werd naar die buur gesprongen met de hoogste activiteitsgraad. Op het laatst werd, wanneer een oplossing mogelijk was, het doel bereikt. Dit model genereerde een pad dat uit een ketting van via-punten bestond.

De bovengenoemde padplanners werden geimplementeerd in de vorm van computer algoritmen en werden verwerkt op sequentiele processoren. Het is duidelijk dat deze methoden slechts weinig bijdragen tot een beter begrip van de moge-

lijke processen in de hersenen die de gegevens juist op een parallelle manier, over vele neuronen, verwerken. En veel vragen moeten nog worden beantwoord. Is het wel mogelijk om neurale activiteiten te interpreteren? Als het mogelijk is, welk coordinatensysteem wordt er dan gebruikt om de omgeving met obstakels en doelen te representeren? Hoe berekent het brein een pad? Hoe bestuurt het de vele spieren die bij een beweging in actie moeten komen? Is het mogelijk om de bovengenoemde sequentiele algoritmen te vertalen in termen van neurale structuren?

Alhoewel reeds een aantal basisprincipes over de werking van het brein bekend is, bestaat er nog geen werkend model van een plausibele neurale padplanner. In deze thesis willen we een bijdrage leveren om dit gat enigszins op te vullen door een aantal specifieke neurale netwerk modellen voor padplanning te beschrijven. De modellen zijn gebaseerd op biologisch plausibele principes over de werking van het brein met daarbij een aantal noodzakelijke aannamen.

Onze modellen worden beschreven in termen van neurale netwerken, i.e. bestaande uit: artificiele neuronen, de verbindingen tussen de neuronen en de dynamica van de neurale activiteiten. Elk neuron is een simpele processor. Het integreert alle inkomende signalen, bewerkt dit volgens een eenvoudige formule en stuurt de uitkomst hiervan naar andere neuronen. Er wordt aangenomen dat het neurale netwerk informatie ontvangt over de objecten in de werkruimte via een aantal speciale sensorneuronen. Om de omgeving intern te representeren heeft het neurale net de gehele werkruimte onderverdeeld in vele kleine subruimten op een manier zoals receptieve velden bij sensorneuronen.

Wanneer de neuronen dusdanig geordend zijn, dat naast elkaar liggende neuronen in het netwerk corresponderen met naast elkaar liggende receptieve velden in de werkruimte, dan wordt het neurale netwerk een topologisch geordende map genoemd (zie figuur 6.2). Deze topografische ordening kan het resultaat zijn van een autonoom adaptief proces in de verbindingen tussen de neuronen. Dit adaptieve proces dat afhangt van de correlaties tussen de sensorische informatie en de neurale activiteiten in de neurale map is een zogenoemde "leerregel". Deze leerregels zijn beschreven in modellen zoals die van Kohonen [39, 59] en in de "vector quantizatie" methoden [59, 17, 49].

Uit de literatuur is bekend dat neuronen niet geisoleerd reageren. Eerder zal een grote groep neuronen worden geactiveerd wanneer een enkele stimulus wordt gegeven. De reagerende neuronen hebben in meer of mindere mate dezelfde eigenschappen en vaak bevinden zij zich in elkaars nabijheid. Het probleem waar vele neurofysiologische onderzoekers op stuiten is hoe de activaties van een populatie neuronen te interpreteren in termen van sensorische informatie of motorische

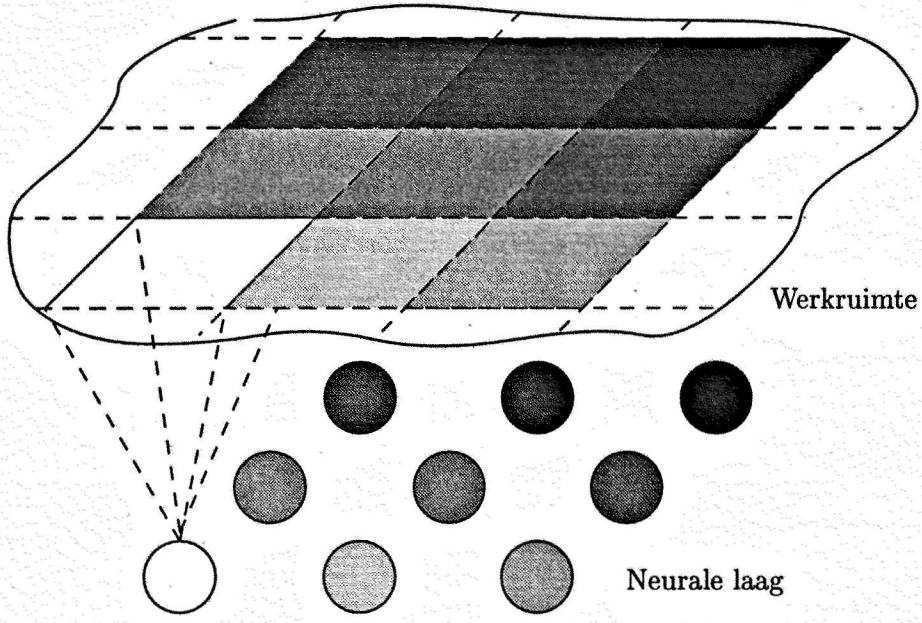


Figure 6.2: De werkruimte is onderverdeeld in subruimten. Dit zijn de receptieve velden van de neuronen. Het neurale netwerk bestaat uit een laag van neuronen. De associatie van neuronen met hun bijbehorende receptieve velden wordt aangegeven met een grijswaarde. In dit geval is de neurale laag een zogenoemde "topografische map" omdat de receptieve velden in de werkruimte op dezelfde manier geordend zijn als de neuronen in de neurale laag.

akties. Georgopoulos et al. [19] hebben een model voorgesteld dat gebaseerd is op de resultaten van experimenteel onderzoek. Dit model bevat een expliciete interpretatie van de activiteiten van een populatie neuronen in de motorcortex in termen van de verplaatsingen van een arm. Een schatting van de bewegingsrichting wordt gemaakt door een sommatie van de voorkeursrichtingen (dat is de bewegingsrichting waarbij de cel de grootste activiteit vertoont) van alle neuronen in de motorcortex, gewogen met de activiteit van de cel. Het resultaat wordt de "populatievector" genoemd.

De populatievector geeft een goede schatting van in neurale activiteiten gecodeerde sensorische of motorische gebeurtenissen op voorwaarde dat de voorkeursrichtingen uniform verdeeld zijn over de neurale map. Het is echter bekend, dat de distributie van de neuronen over de sensorische of motorruimte vaak niet uniform is. Wanneer de populatievector toch onveranderd gebruikt wordt, dan heeft het resultaat van de schatting een afwijking. Gelukkig is het idee van Georgopoulos, in het bijzonder in de vorm van de theorie van Seung en Sompolinsky [68], uit te breiden zodat het gebruik met niet uniform verdeelde maps mogelijk is onder voorbehoud van een aantal noodzakelijke aannamen met betrekking tot de populatie codering. In hoofdstuk vijf bestuderen wij de populatievector in het geval van niet uniform verdeelde neurale netwerken. Een resultaat van dit onderzoek is, dat ons ontwikkelde padontwikkelingsmodel ook met een niet uniform verdeeld topologisch geordend neuraal netwerk kan werken.

Een codering lijkend op de populatievector wordt gebruikt bij de twee neurale netwerk modellen die gepresenteerd worden in hoofdstuk twee. De aanname wordt gemaakt dat de populatie van geactiveerde neuronen continu verandert als het gevolg van de intrinsieke dynamica van het netwerk. De twee neurale netwerken zijn analoog aan het potentiaal algoritme. Het eerste model wordt beschreven door middel van binaire (twee-waardige) neuronen met een stochastisch gedrag. De tweede heeft continu-waardige neuronen en een deterministisch gedrag.

Door gebruik te maken van korte excitoire verbindingen in combinatie met lange inhibitoire verbindingen wordt een stabiel coherent cluster van neurale activiteiten op de map verkregen. Verbindingen vanuit de externe sensorische neuronen zijn aangelegd waarbij de sterkte omgekeerd evenredig toeneemt met de afstand tussen de receptieve velden van het neuron en van het sensorisch neuron. Alle neuronen worden ge-exciteerd behalve die neuronen die de posities van de obstakels coderen. Deze neuronen worden volledig geinhibeerd. Afhankelijk van de doelpositie in de sensorische informatie verplaatst het cluster van activiteiten zich beetje bij beetje over de neurale map in de richting van het doel. Helaas, net zoals bij de veldpotentiaalmethode lijdt dit model aan lokale minima waar het in vast kan komen te zitten.

Het neurale netwerk analogon van de "golfvoortplanting" methode is het onderwerp van hoofdstuk drie. Zoals eerder, gebruiken we een topologisch geordend neuraal netwerk. Deze keer gebruiken we alleen de korte excitoire verbindingen. Externe verbindingen leveren excitoire input naar dat neuron waarbij het doel zich in het receptieve veld bevindt en inhiberen volledig die neuronen die de posities van de obstakels coderen. Wanneer het doelneuron door middel van de externe verbinding geactiveerd wordt, zal het zijn buren, waarmee hij is verbon-

den, activeren. Als de sterkte van de korte dracht verbindingen goed gekozen is zal zich een golf van activiteiten over de map verspreiden. In dit proces zal de activatiewaarde van de neuronen dalen naar gelang het aantal tussenliggende "doorgevende" neuronen in de ketting. Uiteindelijk zal een activatielandschap verrijzen in de vorm van een berg zodat vanuit ieder punt op de map, door middel van aaneengesloten sprongen steeds naar het buurneuron met de hoogste activiteit, een optimaal pad gevormd kan worden naar het doel (de top). Merk op dat deze laatste procedure de aanwezigheid vereist van een toezichthouder, een homunculus, wat zeker niet realistisch is voor een biologisch netwerk. Echter, de methode presteert zeer goed omdat de gegenereerde trajecten optimaal zijn.

Uiteindelijk behalen wij onze doelstelling door het biologisch plausibele model dat in hoofdstuk vier wordt behandeld. In dit model combineren we de twee voorgaande modellen in een twee lagen netwerk. Het netwerk analogon van het golfvoortplantingsmodel vormt de eerste (invoer) laag waarin, als gevolg van externe excitoire input, een activatielandschap wordt gecreeerd en de top met de doelpositie overeenkomt. De neuronen van deze laag zijn met precies een neuron in de tweede laag verbonden. De tweede (motor) laag is analoog aan de veldpotentiaalmethode en bevat een convex cluster van geactiveerde neuronen. Als gevolg van een inhibitoire verbinding tussen de neuronen van de eerste laag met de bijbehorende neuronen in de tweede laag zal het cluster continu en gelijkmatig langs de optimale route naar het doel glijden.

Elke hoofdstuk bevat computersimulaties ter illustratie van de modellen die in het betreffende hoofdstuk worden beschreven.

Een bijkomende eigenschap van het laatste model is dat het gegenereerde pad continu en gelijkmatig is. Als een gevolg hiervan kan ook het motorsysteem continu en gelijkmatig werken. Zou de trajectontwikkelaar alleen zogenoemde "via-punten" genereren, dan is een extra module vereist om als intermediair tussen de trajectontwikkelaar en het motorsysteem te werken. Deze interface moet voorkomen dat de bewegingen gedurende de verplaatsingen van via-punt naar via-punt abrupt en schokkerig zijn.

Vanwege het lage aantal verbindingen in het laatste model tussen de neuronen en vanwege het feit dat zowel de dynamica als de niet-lineaire input-output relatie van de neuronen continu zijn, is het mogelijk om dit laatste model in analoge elektronische hardware te implementeren. Dat wil zeggen zonder digitale elementen. De parallelle niet-digitale verwerkingstijd in een dergelijke elektronische padontwikkelaar is een aantal factoren kleiner dan de bewegingstijd van de robot.

Het grootste nadeel gedurende dit onderzoek was dat de laatste padontwikke-

laar gesimuleerd moest worden op een sequentiele computer. Het groot aantal berekeningen in de algoritmen veroorzaakten relatief lange berekeningstijden. Desalniettemin, was het golfvoortplanting neurale netwerk als een sequentieel algoritme snel genoeg om voor real-time robotica gebruikt te worden. In samenwerking met de Universiteiten van Amsterdam en Utrecht werd dit model met succes gebruikt in een demonstratieproject waarbij het besturen van een robotarm met 6 vrijheidsgraden door middel van alleen maar neurale netwerken het doel was.

Bibliography

[1] L.F. Abbott. Decoding neuronal firing and modeling neural networks. *Q. Rev. Biophysics*, 27:291–331, 1994.

[2] S. Amari. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27:77–87, 1977.

[3] J. Barraquand, B. Langlois, and J.C. Latombe. Numerical potential field techniques for robot path planning. *IEEE Transactions on Systems, Man, and Cybernetics*, 22:224–241, 1992.

[4] J. Barraquand and J.C. Latombe. Robot motion planning with many degrees of freedom and dynamic constraints. In *Proceedings of the 5th International Symposium on Robotics Research (Tokyo).*, pages 74–83, 1989.

[5] J. Barraquand and J.C. Latombe. A monte-carlo algorithm for path-planning with many degrees of freedom. In *Proceedings of the IEEE International Conference on Robotics and Automation (Cincinnati, OH).*, pages 1712–1717, Los Alamitos, 1990. IEEE Computer Society Press.

[6] W. Bialek and F. Rieke. Reliability and information transsmition in spiking neuron. *Trends in Neurosciences*, 15:428–433, 1988.

[7] E. Bizzi, N. Accornero, N. Chapelle, and N. Hogan. Arm trajectory formation in monkeys. *Experimental Brain Research*, 46:139–143, 1982.

[8] E. Bodewig. *Matrix Calculus.* North-Holland, Amsterdam, 1959.

[9] R. Caminiti, P. B. Johnson, Y. Burnod, C. Galli, and Ferraina S. Shifts of preferred directions of premotor cortical cells with arm movements performed across the work-space. *Experimental Brain Research*, 83:228, 1990.

[10] M. A. Cohen and S. Grossberg. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:815, 1983.

[11] C.I. Connolly, J.B. Burns, and R. Weiss. Path planning using Laplace's equation. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2102–2106, Los Alamitos, 1991. IEEE Computer Society Press.

[12] J.L. Crowley. Navigation for an intelligent mobile robot. *IEEE Journal of Robotics and Automation*, RA-1:31, 1985.

[13] L. Dorst, I. Mandhyan, and K. Trovato. The geometrical representation of path planning problems. *Robotics and Autonomous Systems*, 7:181, 1991.

[14] J. Droulez and A. Berthoz. A neural network model of sensoritopic maps with predictive short term memory properties. *Proceedings of the National Academy of Sciences of the United States of America*, 88:9653, 1991.

[15] D. C. van Essen, W. T. Newsome, and J. H. R. Mannsell. The visual field representation in striate cortex of the macaque monkey: assymtries, anisotropies and individual variability. *Vision Research*, 24:429–448, 1984.

[16] A. G. Feldman. Once more on the equilibrium-point hypothesis (λ model) for motor control. *Journal of Motor Behavior*, 18:17–54, 1986.

[17] B. Fritzke. Unsupervised clustering with growing cell structures. In *Proceedings of the International Joint Conference on Neural Networks (Seatle/WA).*, volume II, pages 531–536, Piscataway, NJ, 1991. IEEE.

[18] A.P. Georgopoulos, J. Ashe, N. Smyrnis, and M. Taira. The motor cortex and the coding of force. *Science*, 256:1692–1695, 1992.

[19] A.P. Georgopoulos, J.F. Kalaska, M. D. Crutcher, R. Caminiti, and J.T. Massey. The representation of movement direction in the motor cortex: single cell and population studies. In W.M. Cowan G.M. Edeiman, W.E. Gail, editor, *Dynamic aspects of neocortical function.*, pages 501–524. Neurosciences Research Fundation, Inc., 1984.

[20] A.P. Georgopoulos, R.E. Kettner, and A. Schwartz. Neural population coding of movement directions. *Science*, 243:234–236, 1986.

[21] C.C.A.M. Gielen, R. Glasius, and A. Komoda. Interpretation of neuronal activity in neural networks. *Journal of Neurocomputing.*, In press, 1996.

[22] C.C.A.M. Gielen, G.H.F.M. Hesselmans, and P.I.M. Johannesma. Sensory interpretation of neural activity patterns. *Mathematical Biosciences.*, 88:15–35, 1988.

[23] R. Glasius, A. Komoda, and C.C.A.M. Gielen. Population coding in a neural net for trajectory formation. *Network, Computation in Neural Systems*, 5:549–563, 1994.

[24] R. Glasius, A. Komoda, and C.C.A.M. Gielen. Neural network dynamics for trajectory formation and obstacle avoidance. *Neural Networks*, 8:125–133, 1995.

[25] R.J. Glauber. Time-dependent statistics of the Ising model. *Journal of Mathematical Physics.*, 4:294–307, 1963.

[26] H. P. Graf, L. D. Jackel, R. E. Howard, B. Straughn, J. S. Denker, W. Hubbard, D. M. Tennant, and D. Schwartz. VLSI implementation of a neural network memory with several hundreds of neurons. In J. S. Denker, editor, *Neural Networks for Computing (Snowbird 1986).*, page 182, New York, 1986. American Institute of Physics.

[27] S. Grossberg. Nonlinear neural networks: principles, mechanisms, and architectures. *Neural Networks*, 1:17, 1988.

[28] J. Heikonen, P. Koikkalainen, and E. Oja. From situations to actions: motion behavior learning by self-organization. In *Proceedings of the International Conference on Artificial Neural Networks (Amsterdam/The Netherlands)*, pages 262–267, Amsterdam; North-Holland, 1993. Springer-Verlag.

[29] N. Hogan. An organizing principle for a class of voluntary movements. *The Journal of Neural Science*, 4:2745–2754, 1984.

[30] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79:2554, 1982.

[31] J. J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences of the United States of America*, 81:3088, 1984.

[32] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *Journal of Physiology*, 160:106–154, 1962.

[33] R. A. Jarris. Collission-free trajectory planning using distance transforms. *Mech Eng Trans of the IE Aust.*, ME10:187, 1985.

[34] J. F Kalaska, D. Crammond, D. A. D. Cohen, Prud'homme M., and M. L. Hyde. *Comparision of cell discharge in motor, premotor and parietal cortex during reaching. Load direction-related activity in primate motor cortex, using a two dimensional reaching task.* Springer Verlag, New York, 1992.

[35] K. Kant and S.W. Zucker. Towards efficient trajectory planning: The path-velocity decomposition. *The International Journal of Robotics Research*, 5:72–89, 1986.

[36] O. Kathib. Real-time obstacle avoindance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5:90–98, 1986.

[37] N. Kiang. *Discharge patterns of single fibers in the cat's auditory nerve.* M.I.T. Press, Cambridge, 1965.

[38] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[39] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[40] K. Kopecz and G. Schöner. Saccadic motor planning by integrating visual information and pre-information on neural dynamic fields. *Biological Cybernetics*, 73:49–60, 1995.

[41] B.H. Krogh. A generalized potential field approach to obstacle avoidance control. In *Proceedings of the IEEE International Conference on Robotics and Automation*, page 948, Los Angeles, 1984. Computer Society Press of the IEEE.

[42] B.H. Krogh and C.E. Thorpe. Integrated path planning and dynamic steering control for autonomous vehicles. In *Proceedings of the IEEE International Conference on Robotics and Automation (Washington DC)*, pages 1664–1669, Los Angeles, 1986. Computer Society Press of the IEEE.

[43] J.C. Latombe. *Robot motion planning.* Kluwer Academic, Boston, 1991.

[44] R. Linsker. From basic network principles to neural architecture: emergence of orientation-selective cells. *Proceedings of the National Academy of Sciences of the United States of America*, 83:8390–8394, 1986.

[45] R. Linsker. From basic network principles to neural architecture: emergence of orientation columns. *Proceedings of the National Academy of Sciences of the United States of America*, 83:8779–8783, 1986.

[46] R. Linsker. From basic network principles to neural architecture: emergence of spatial-opponents cells. *Proceedings of the National Academy of Sciences of the United States of America*, 83:7508–7512, 1986.

[47] T. Lozano-Perez. Spacial planning: a configuration space approach. *IEEE Transactions on Computers*, C-32:108–120, 1983.

[48] C. M. Marcus, F. R. Waugh, and R. M. Westervelt. Associative memory in an analog iterated-map neural network. *Physical Review A*, 41:3355, 1990.

[49] T. M. Martinetz. Competitive hebbian learning rule forms perfectly topology preserving maps. In *Proceedings of the International Conference on Artificial Neural Networks (Amsterdam/The Netherlands)*, pages 427–434, Amsterdam; North-Holland, 1993. Springer-Verlag.

[50] J. T. McIlwain. Visual receptive fields and their images in suprior colliculus of the cat. *Journal of Neurophysiology*, 20:535–538, 1975.

[51] P. Morasso, V. Sanguineti, and T. Tsuji. Neural architecture for robot planning. In *Proceedings of the International Conference on Artificial Neural Networks (Amsterdam/The Netherlands)*, pages 256–261, Amsterdam; North-Holland, 1993. Springer-Verlag.

[52] D.P. Munoz, D. Pelisson, and D. Guitton. Movement of neural activity on the superior colliculus motormap during gaze shifts. *Science*, 251:1358–1360, 1991.

[53] W.S. Newman and N. Hogan. High speed robot control and obstacle avoidance using dynamic potential function. In *Proceedings of the IEEE International Conference on Robotics and Automation (Raleigh, NC)*, pages 14–24, Los Angeles, 1987. Computer Society Press of the IEEE.

[54] L. Peichle and H. Wässle. Size, scatter and coverage of ganglion cell receptive dield centres in the cat retina. *Journal of Physiology*, 291:117–141, 1979.

[55] L. Peichle, H. Wässle, U. Grünert, J. Röhrenbeck, and B.B. Boycott. Retinal ganglion cell density and cortical magnification factor in the primate. *Vision Research*, 30:1897–1911, 1990.

[56] C. Peterson and J. R. Anderson. A mean field theory learning algorithm for neural networks. *Complex Systems*, 1:995–1019, 1987.

[57] R. Porter and R. Lemon. *Corticospinal Neurones. Their Role in Movement.* Academic., London, 1977.

[58] E. Prassler. Electrical networks and a connectionist approach to path-finding. In *Connectionism in Perspective*, page 421, Amsterdam, 1989. North-Holland.

[59] H. J. Ritter, T. M. Martinetz, and K. J. Schulten. Topology-conserving maps for learning visuo-motor-coordination. *Neural Networks*, 2:159–168, 1989.

[60] H. J. Ritter, T. M. Martinetz, and K. J. Schulten. *Neural computation and self-organizing maps. An introduction.* Addison-Wesley Publ. Comp., 1992.

[61] D. A. Robinson. Eye movements evoked by collicular stimulation in the alert monkey. *Vision Research*, 12:1795–1808, 1972.

[62] M. G. P. Rosa and l. M. Schmid. Magnification factors, receptive field images and point-image size in the superior colliculus of flying foxes: comparison with the primary visual cortex. *Experimental brain research*, 102:551–556, 1995.

[63] J. Rovamo and V. Virsu. Isotropy of cortical magnification and topography of striate cortex. *Vision Research*, 24:283–286, 1984.

[64] E. Salinas and L. F. Abbott. Vector reconstruction from firing rates. *Journal of Computational Neuroscience*, 1:89–107, 1994.

[65] E. L. Schwartz. Computational anatomy andfunctional architecture of striate cortex: A spatial mapping approach to perceptual coding. *Vision Research*, 20:645–669, 1980.

[66] J. T. Schwartz and M. Sharir. On the "piano" movers problem. II General techniques for computing topological properties of real algebraic manifolds. *Advances in Applied Mathematics*, pages 298–351, 1983.

[67] C. Seshadri and A. Ghosh. Optimum path planning for robot manipulators amid static and dynamic obstacles. *IEEE Transactions on Systems, Man, and Cybernetics*, 23:576–584, 1993.

[68] H. S. Seung and H. Sompolinsky. Simple models for reading neuronal population codes. *Proceedings of the National Academy of Sciences of the United States of America*, 90:10749–10753, 1993.

[69] D. L. Sparks. Neural cartography: Sensory and motor maps in the superior colliculus. *Brain Behavior and Evolution*, 31:49–56, 1988.

[70] P. P. van der Smagt. Minimisation methods for training feed-forward networks. *Neural Networks*, 7:1–11, 1994.

[71] P. P. van der Smagt and B. J. A. Krose. A real-time neural robot controller. In *Proceedings of the International Conference on Artificial Neural Networks (Espoo/Finland)*, pages 351–356, Amsterdam; North-Holland, 1991. Elsevier Science Publishers.

[72] J.A.M. van Gisbergen, A.J. van Opstal, and A.M.M. Tax. Collicular ensemble coding of saccades based on vector summation. *Neuroscience.*, 21:541–555, 1987.

[73] J. M. Vleugels, J. N. Kok, and M. H. Overmars. A self-organizing neural network for robot motion planning. In *Proceedings of the International Conference on Artificial Neural Networks (Amsterdam/The Netherlands)*, pages 256–261, Amsterdam; North-Holland, 1993. Springer-Verlag.

[74] Y. Wada and M. Kawato. A neural network model for arm trajectory formation using forward and inverse dynamics models. *Neural Networks*, 6:919–932, 1993.

[75] C.W. Warren. Global path planning using artificial potential fields. In *Proceedings of the IEEE International Conference on Robotics and Automation (Scottsdale, AZ)*, pages 316–321, Los Angeles, 1989. Computer Society Press of the IEEE.

[76] C.K. Yap. *Algorithmic motion planning.* Hillsdale, N.J., Erlbaum, 1986.

Publications

R. Glasius, A. Komoda, C.C.A.M. Gielen, "The Population Vector, an unbiased estimator for non-uniformly distributed neural maps", *Accepted by Neural Networks*, 1997.

C.C.A.M. Gielen, R. Glasius, A. Komoda, "Interpretation of neuronal activity in neural networks." *Journal of neurocomputing*, 1996, vol. 12, pp. 249-266.

R. Glasius, A. Komoda, C.C.A.M. Gielen, "A Biologically inspired Neural net for trajectory Formation and Obstacle avoidance.", *"Biological Cybernetics*, 1996, vol. 84, pp. 511-520.

R. Glasius, A. Komoda, C.C.A.M. Gielen, "Neural Network Dynamics for Path Planning and Obstacle Avoidance" *Neural Networks*, 1995, Vol. 8, No. 1, pp. 125-133

R. Glasius, A. Komoda, C.C.A.M. Gielen, "Population Coding in a Neural Net for Trajectory Formation" *Network, Computation in Neural Systems*, 1994, Vol 5, 549-563.

R. Glasius, A. Komoda, C.C.A.M. Gielen, "Biologically Inspired Neural Network for Trajectory Formation and Obstacle Avoidance.", Proc. ICANN'93 Conference Amsterdam, Springer Verlag, London, 1993, 646-649,

B. Krose, M. Bartholomeus, A. Noest, R. Glasius, A. Komoda. "Neural networks for real-time robotics and vision applications.", video track session, ECAI'94 Conference, 8-12 August 1994, Amsterdam

C.C.A.M. Gielen, R. Glasius, "Coordination of Movements by Self-Organizing Neural Netorks" IAS-2 Conference, 11-14 December 1989, Amsterdam.

Dankwoord

Het produceren van een proefschrift en het verkrijgen van de daarin beschreven inzichten, is een proces met een zeer hoge viscositeitsgraad. Gelukkig hoeft de promovendus de talloze voetangels en hindernissen niet alleen te nemen. Volgens goed gebruik wil ik al diegenen bedanken die een bijdrage hebben geleverd bij het tot stand komen van dit werk.

Vanzelfsprekend is daarbij mijn promotor Stan Gielen, die ik graag wil bedanken voor de kans die hij mij heeft gegund.

Mijn directe begeleider Andrzej Komoda. Naast zijn vriendschap heb ik van hem die hulp verkregen, zonder welke ik in alle stroperigheid van het proces zou zijn blijven steken en van dit boekje niets terecht was gekomen.

Al diegenen die niet direct met het werk te maken hadden, maar mijn leven in de vakgroep in meer of mindere mate veraangenaamd hebben. Dat zijn mijn talloze kamergenoten, de volleybalspelers (totdat ik mijn enkelbanden meerdere keren achter elkaar verrekt heb), de lieden van het queeste-spel annex eetfeest annex roddelgroep (hopelijk "a continuing story"), alle DOOM tegenspelers en diegenen waarbij ik mijzelf regelmatig voor het eten heb mogen uitnodigen (en vice versa). Hierbij wil ik speciaal hen noemen die zich in deze hebben onderscheiden en dus de twijfelachtige eer hadden om zich meer dan gemiddeld in mijn nabijheid te frequenteren. Dat zijn Ger van Lingen, Siebren Schaafsma, Thom Oostendorp en Marc Albers.

Verder bedank ik Victor Langeveld en Gunter Windau, leden van de computer groep, waarvan ik op Unix/Linux gebied het een en ander heb mogen opsteken.

Natuurlijk moet ik ook mijn familie noemen waar ik mij ieder weekend in de nestwarmte mag koesteren om zo steeds weer op te laden voor een volgende werkweek (dit onder andere door middel van een (on)gezonde hoeveelheid Indisch eten).

Wellicht is dit niet de juiste plaats om Monica, Mick en Alec te noemen. Hun aanwezigheid in mijn leven heeft, meer dan de oorzaak van een niet uit te drukken dank, rechtstreeks te maken met mijn zin van/in het leven.

Curriculum Vitae

Op 15 maart 1961 ben ik te Bandung op het eiland Java (Indonesie) midden in de "Gordel van Smaragd" geboren. Op een koude winterdag in December 1964 kwamen wij per vliegtuig in Nederland aan. Mijn vaders eerste stap vanaf de vliegtuigtrap op de ijzige vaderlandse bodem is wellicht illustratief voor het verschil en de onbekendheid met de Hollandse cultuur. Ondanks de buiteling die hij destijds maakte, hebben wij uiteindelijk ons plekje in deze maatschapij kunnen vinden.

Voor mij bestond deze tocht uit een opleidingstraject dat zich geheel in Nijmegen heeft afgespeeld.

Ik begon op vijfjarige leeftijd op de kleuterschool "Kabouterland" waarbij ik in het overgangsrapport een tien scoorde voor "vechten" en "ondeugendheid". Vervolgens bezocht ik gedurende zes jaren de lagere school "Bisschop Beckers" om van 1973 tot 1978 de HAVO te doen van de scholengemeenschap "Dukenburg". In 1982 deed ik het eindexamen Atheneum-B op het "Dominicus college" welke mij toegang gaf tot de studie natuurkunde aan de "Katholieke Universiteit Nijmegen". De propaedeuse behaalde ik in 1983 en het Doctoraal in 1988. Mijn afstudeerstage bij de vakgroep "Molecuul en Laser Fysica" onder leiding van professor Reuss bestond uit het ontwikkelen en bouwen van een "short wave guide CO_2" laser. Nadien ben ik terecht gekomen bij de vakgroep Biofysica van professor Gielen waarbij ik tot november 1996 gewerkt hebt. Deels onder de mantel van de "Stichting Neurale Netwerken" heeft mijn onderzoek al daar geleid tot deze promotie.

Per 1 november 1996 ben ik begonnen als systeemontwikkelaar bij TAS Informatica te Baarn. Vanaf 1 December 1997 zal ik hetzelfde beroep uitoefenen bij IP/Informatica Projectgroep in Diemen.